# المتحكمات الإلكترونية

الكتاب الشامل

ECTRONICS ENGINEERING MAGAZINE

هذا الكتاب يشتمل على الآتى :

- الميكروكنترولر MicroController
- الأردوينو Arduino
- الراسبيرى باى Raspberry Pi
- التحكم المنطقى المبرمج PLC


اعداد

م. فوزى الأزرق


المتحكمات الإلكترونية - الطبعة الأولى

# المتحكمات الإلكترونية

فوزي عبدالله الأزرق

4

# المتحكمات الالكترونية

اعداد : م.فوزي الازرق

الطبعة الاولى – النسخة التجريبية

المملكة الاردنية الهاشمية

رقم الايداع لدى دائرة المكتبة الوطنية

(2019/8/4341)

621,39

الازرق, فوزي عبدالله

المتحكمات الالكترونية / فوزي عبدالله الازرق .-عمان:المؤلف,2019

()ص.

ر.إ: 2019/8/4341 .

الواصفات : /هندسة البرمجيات//الدوائر الالكترونية/هندسة الحواسيب/

# رخصة الكتاب

# عن الكاتب

الازرق , ولد عام 1997 في عمان – الاردن ,التحق في الدراسة الثانوية الصناعية بمسار الاتصالات والالكترونيات في عام 2013 , في عام 2015 درس مرحلة البكالريوس في الجامعة الهاشمية في كلية الهندسة تخصص الهندسة الميكانيكية العامة , اهتم في علوم الهندسة الميكانيكية والكهربائية والذكاء الاصطناعي وعلم الحاسوب وعلوم الطيران , ومن هواياته ركوب الدراجات النارية والطيران الشراعي والرماية والصيد . اهتم في الثقافات وكتابة الكتب الهندسية والبرامج الهندسية باستخدام البرامج المتطورة , بعض من كتبه :

- كتاب الماتلاب للمهندسين والدراسات الاقتصادية والمالية.
- كتاب الديناميكا الهوائية وتصميم الطائرات.
- كتاب برمجة محركات السيارات.
- كتاب Introduction to ICE and Design using Matlab and AI
- كتاب Introduction to Engineering programing
- كتاب ماذا استطعت ان اعلم نفسي في 24 ساعة.

يعيش الازرق في حياة مليئة في اجواء من الاختراعات والتعلم

الازرق كرس حياته في التعلم , وتعليم الناس ..

# اهداء السلسلة

إن الحمد لله، نحمدُه ونستغفره ونستعينه ونستهديه ونعوذُ بالله من شرورِ أنفسنا ومن سيئاتِ أعمالنا، من يهْدِ اللهُ فلا مضِلَّ له ومن يضلل فلا هادي له.

احمدُ الله تعالى على هذا الإنجاز الجديد وهي سلسلة الذكاء الاصطناعي والانظمة الميكانيكية والتحكم الالكتروني وهذا النجاح احب مشاركته واهدائُه, فأُهدي هذه السلسلة بكتبها البالغ عددها 10 كتب والذي ألهمنيَ الله تعالى في كتابتها

إلى عائلتي وأخص بالذكر أبي والراحلة

والدتي الحبيبة رحمها الله ..

واني اوجه الشكر الى كل معلم علمني حرف واحدًا فلولاكم لما وصلت لهذه المرحلة , واسال الله ان يقدرنا على اثراء المحتوى العربي وتعليم الشباب العرب كل ما ينفعهم.

# مقدمة السلسلة

بسم الله الرحمن الرحيم

جاء في القرآن الكريم بعد اعوذ بالله من الشيطان الرجيم

﴿ إِنَّمَا يَخْشَى اللَّهَ مِنْ عِبَادِهِ الْعُلَمَاءُ إِنَّ اللَّهَ عَزِيزٌ غَفُورٌ ﴾ [فاطر: 28]

وعن أبي هريرة رضي الله عنه قال قال صلى الله عليه وسلم (إذا مات ابن ادم انقطع عمله إلا من ثلاث : صدقة جاريه , او علم ينتفع به , أو ولد صالح يدعوا له) أخرجه مسلم.

وجاء في سفر الأمثال الإصحاح 18 و العدد 15 (قَلْبُ الْفَهِيمِ يَقْتَنِي مَعْرِفَةً، وَأُذُنُ الْحُكَمَاءِ تَطْلُبُ عِلْمًا).

عندما وُلدنا — من وقت غير بعيد — لم تكن تكنولوجيا المعلومات أوشركات التليفزيون موجودة، وكان السفر الجوي نادرًا وضربًا من الرفاهية. أما آباؤنا، فقد وُلدوا في عالم أكثر اختلافًا عن عالمنا الحالي؛ فلم يكن التليفزيون قد اختُرع بعد، ولم يكن يوجد أطعمة مجمدة. وعندما وُلد أجدادنا، لم تكن توجد محركات احتراق داخلي أوطائرات أو دور سينما أو أجهزة راديو، وعاش آباء أجدادنا في عالم لا توجد فيه مصابيح ضوئية أو سيارات أو هواتف أو دراجات أو ثلاجات أو آلات كاتبة، وربما كانت حياتهم أقرب لحياة الفلاح الروماني عن قربها من حياتنا اليوم. وخلال فترة المائة والخمسين عامًا الماضية والقصيرة نسبيًا، تحولت حياتنا تمامًا سواءٌ في المنزل أو العمل بواسطة المنتجات والخدمات الجديدة. والسبب في تغير العالم تغيرًا كبيرًا يمكن تفسيره بدرجة كبيرة بالابتكار. ومن هذا المنطلق ورغبة مني وطلبًا من اصحاب العلم , اطرح سلسلة كتبي الجديدة بعنوان سلسلة الذكاء الاصطناعي والانظمة الميكانيكية والتحكم الالكتروني والتي تشمل مواضيع اساسية في العلوم الهندسية والفيزيائية والبرمجة والتحكم الآلي والطيران وهذا بواقع 10 كتب بالعناوين التالية:

- بناء التطبيقات الهندسية
- الروبوتات
- طائرات التحكم عن بعد
- الماتلاب
- الذكاء الاصطناعي - البرمجة المتقدمة
- تصميم مختبر الالكترونيات
- اساسيات في الالكترونيات والمستشعرات
- مقدمة في البرمجة
- المتحكمات الالكترونية
- التحكم عن بعد

تتناول هذه السلسلة المحاور التكنولوجية المهمة التي يبحث عنها المهندس والباحث والهواة وكل من يهتم في هذا المجال فنحن نعيش في ثورة صناعية وعولمة متسارعة

النمو على الصعيد التقني , تتحدث هذه السلسلة بشكل تدريجي ومبسط عن اهم المواضيع القديمة والحديثة في مجال الذكاء الاصطناعي والمتحكمات الالكترونية والمعالجات والطائرات والتحكم والالكترونيات والصناعة ,في هذه السلسلة قدمت لكم حزمة معلومات لكي تبدأ بها كحجر الاساس لك لتواكب وتركب امواج التكنولوجيا . ومن واجبنا السعي قدما لتعلم هذه العلوم حتى لا نقع في ظلم الجهل ونصبح شعوب مستهلكة ضعيفة لا قدر الله!

بدئت الدروس في كتاب تصميم مختبر الالكترونيات , والذي يتناول بماذا يلزمنا لبناء المختبر واهم الاجهزة والادوات ومواصفاتها الفنية , اضافة الى كيفية التعامل معها للبدء بداية مثالية فدائما اقول (من لم تكن له بداية محرقة لن تكن له نهاية مشرقة). ثم انتقلت الى مرحلة التأسيس في مجال الالكترونيات والمحركات والمستشعرات المختلفة في كتاب اساسيات في الالكترونيات والمستشعرات بحيث يتناول هذا الكتاب معظم العناصر الالكترونية من مقاومات وملفات ومكثفات وثنائيات وغيرها وايضا طريقة تصنيعها واعطالها والخصائص الفيزيائية لها وامثلة حسابية من واقع الحياة ودروس في القوانين الكهربائية مثل التيارات المسمرة والمتناوبة وقانون اوم وغيرها .

ثم انتقلت الى البرمجة في كتاب مقدمة في البرمجة , تحدثت عن لغات البرمجة ومصطلحات في البرمجة وتطرقنا الى حل المشاكل واستخدام الخوارزميات ومخططات سير العمليات وبدئنا في كتابة البرامج باستخدام لغة سي بلس بلس حيث اندرجت المواضيع من المتغيرات والثوابت والادخال والاخراج والجمل الشرطية والدورانية المختلفة , اضافة الى المصفوفات وسبب اختيار لغة سي بلس بلس هو اننا سوف نستخدمها في السلسلة لبرمجة المتحكمات الالكترونية والماتلاب لذلك ينبني عليها الكثير من الاعمال المهمة .

بعد ان تعلم القارئ البرمجة واساسيات الالكترونيات اصبح قادرا على البدء في دراسة المتحكمات الالكترونية قدمت في كتاب المتحكمات الالكترونية دروس يستطيع من خلالها المتعلم فهم اساس علم المتحكمات الالكترونية القابلة والغير قابلة للبرمجة والمعالجات المختلفة تناولت دروس عن الدوائر المتكاملة والدوائر المنطقية والمعالجات الدقيقة والميكروكونترولر ودخلت الى متحكم الاردوينو انواعه واستخداماته وكيفية برمجته وتنصيب احتياجاته والتعامل معه اضافة الى اساسيات في الراسبيري باي والمتحكم المنطقي القابل للبرمجة (PLC).

ثم انتقلت الى دراسة تقنيات التحكم عن بعد في كتاب التحكم عن بعد يتحدث هذا الكتاب عن بعض التقنيات مثل البلوتوث وال IR والوايرلس GPS وانظمة الملاحة وهذا لان التحكم عن بعد من المواضيع الهامة في مشاريع هذا الكتاب والمشاريع العملية في الحياة الصناعية . بعد ذلك اصبح من اولوياتي ان اجعل التعليم ممتعا فطرحت الكثير من المشاريع التفاعلية في كتاب بناء التطبيقات الهندسية يضم مشاريع كثيرة مثل اذرع الروبوتات والمنزل الذكي والمشاريع التفاعلية والمنبهات واستخدام التحكم عن طريق الهاتف وغيرها من المشاريع المتنوعة باستخدام الاردوينو والعناصر الالكترونية والدوائر المتكاملة والحساسات والمحركات ..الخ

بعد ان اصبح المتعلم ملما في الكثير من المواضيع الهندسية والبرمجية اصبح قادرا على تعلم تصميم الروبوتات المختلفة فوضعت في كتاب الروبوتات عدة امثلة لروبوتات منزلية وترفيهية تقوم باعمال مختلفة فبعد اجتياز هذه السلسلة يستطيع المتعلم تطوير روبوت خاصة به

ويقوم بالمهام حسب ما يراه المتعلم مناسبا, ثم انتقلنا الى تصميم الطائرات في كتاب مستقل باسم طائرات التحكم عن بعد في هذا الكتاب طرحت مواضيع في الديناميكا الهوائية وقوانين خاصة بالطائرات واستقرارها وانواع الطائرات المختلفة وكيف بناؤها ومحركاتها والكثير من الدروس الخفيفة وسهلة الفهم فهذا الكتاب من احدى الكتب الاكثر طلبًا لدى الشباب العرب

وفي آخر كتابين من السلسلة بدئت في دروس متخصصة في مجال الماتلاب والمحاكاة باستخدم سيميلوينك الاول باسم الماتلاب ويحتوي هذا الكتاب اساسيات البرمجة في الماتلاب واستخدام العمليات الرياضية والمنطقية ورسم المنحنيات والمتغيرات والثوابت والشرط والدوران والمصفوفات ومكتبة سيميولينك للمحاكاة.

واخيرا كتاب متقدم يحمل اسم الذكاء الاصطناعي - البرمجة المتقدمة في هذا الكتاب وضعت دروسا هامة لتعريف المتعلمين عن اهمية الذكاء الاصطناعي ومستقبله المشرق فيضم الكتاب البرمجة باستخدام المنطق الضبابي والشبكات العصبونية واساسيات في الرؤية الحاسوبية , اذ يستطيع المتعلم توظيف الذكاء الاصطناعي في اعماله .

اصدار الجزء الاول هذه السلسة في عام 2019 , في كل عامين او ثلاثة اعوام يقدم الكاتب نسخة جديدة لكي تبقى مواكبة للتكنولوجيا في كل زمان ومكان.

عزيزي المتعلم هذه السلسلة ليست الا البداية فهذه العلوم من العلوم الكبيرة , قد اكون اخطئت او سهوت وقد اكون قصرت في الشرح لذلك يتوجب عليك ان تعلم ان هذا الكتاب ليس معصوما من الخطأ , فيوجد عشرات الكتب التي تتحدث عن المواضيع التي تناولتها وكل كتاب به ما يميزه عن غيره .

عليك دائما بالبحث عن المعلومات وعدم الملل والكلل والتوسع بهذه العلوم , فهذه السلسلة هي خطوتك الاولى لهذا العلم ولا تترك درسا الا ان تكون قد تعلمته جيدا .

(وتذكر ان تصل متأخرًا خيرًا من ان لا تصل)

# مقدمة الجزء

في هذا الجزء من السلسلة يتحدث كتاب المتحكمات الالكترونية عن السلالة الزمنية لعصر التكنولوجيا بدئت الدروس من تطور المتحكمات الالكترونية ودرسنا انواع الاشارات الكهربائية وقمنا في تفصيل لدراستنا لعلم انظمة العد ودخلنا عالم قديم من علوم الالكترونيات وهو الدوائر المتكاملة ذلك الاختراع الذي غير حجم الحواسيب من غرف ومخططات كبيرة الى قطعة سوداء تتكحل بها الاجهزة الالكترونية لتصبح اكثر جمالًا في حجمها وادائها ودرسنا بعض انواع الدوائر المتكاملة مثل 555 ومضخم العمليات وقمنا بتنفيذ بعض المشاريع البسيطة , وبدئنا في التسلسل الزمني لعلوم المتحكمات الالكترونيات حيث انتقلنا الى دراسة الدوائر المنطقية وبدئنا في دراسة المتحكمات الالكترونية القابلة للبرمجة من الصفر وهو متحكم الميكروكنترولر , ليكون حجر الاساس للمتحكم العصري الجديد الاردوينو , وهو هدفنا الاساسي من هذا الكتاب ان نفهم وندرس الاردوينو بشكل كاف لبناء اي نظام الكتروني ذكي بتكلفة صغيرة وامكانيات تكنولوجية هائلة !

بدئت دروس الاردوينو بالتعرف على مكوناته وللغة برمجته وطريقة استخدام اوامره الخاصة وبناء نماذج اولية باستخدامه , وتعرفنا على كيفية تطبيق الجمل الشرطية والدورانية في برمجة المتحكمات الالكترونية القابلة للبرمجة , تعرفنا في تفاصيل على مكونات الاردوينو من حيث البنية الداخلية والمداخل الرقمية والتماثلية ويغرها التي يوفرها وبالتأكيد درسنا المستعشرات والشاشات والمحركات والاغطية وكيفية التحكم عن بعد مع امثلة عملية كبيرة , ثم انتقلنا الى المتحكم الالكتروني العملاق القابل للبرمجة وهو الراسبري باي , ذلك الحاسوب الجبار الذي تستطيع به بناء حاسوب شخصي او التحكم في طائرات وانظمة ذكية جدا كل هذا باستخدام للغة بسيطة درسناها منذ الصغر وهي البايثون ولكن درسنا الراسبيري باي باختصار في الاصدار التجريبي .

وفي نهاية الكتاب خصصنا جزء للمصانع وهو ال PLC للتحكم في منظومة ادارة المصنع آليًا فدرسنا انواع ال PLC وكيفية برمجته وكيفية تطبيقه للمصانع والانظمة الآلية دون الحاجة الى تكاليف كبيرة , دروس ال PLC غير متوفرة في الاصدار التجريبي لعام 2019 .

هذا الكتاب به مجموعة ضخمة من المشاريع الالكترونية ويستهدف المهندسين والهواة وكل من يحب هذه العلوم الممتعة , هذا الكتاب من الكتب المهمة في السلسلة لبناء انظمة الذكاء الاصطناعي والطائرات الاسلكية , يمكنك بعد الانتهاء من هذا الكتاب بناء اي نظام الكتروني قد سبق ان رأيته او فكرت به فقد يكون روبوتات / منازل ذكية / حواسيب شخصية / طائرات بلا طيار / اجهزة طبية / انظمة تحكم / سفن / غواصات / سيرفرات / ادارة مصانع / هندسة عكسية / منظومة ذكاء اصطناعي ... الخ

ان المبدأ هو واحد في جميع المتحكمات الالكترونية القابلة والغير قابلة للبرمجة , ولكن عليك ان تجتهد وتبحث لترى ثمرة جهدك...

# شكر لكل من ساهم في إثراء السلسلة

اتقدم بالشكر الى كل من ساهم في اثراء هذ الجزء من السلسلة (المتحكمات الالكترونية) من الناطقين باللغة العربية والانجليزية والمعاهد فكانت المعلومات والدروس التي قدموها اسهمت في تحقيق اهداف السلسلة وكان لا بد من ذكرهم :



















tutorialspoint
SIMPLYEASYLEARNING


4 ELECTRON
عالم الإلكترون
عالم المستقبل


obital


ElectronicsTutorials


EL-PRO-CUS
ELECTRONICS | PROJECTS | FOCUS


مستقل


آردوينو ببساطة
تعلم الإلكترونيات التفاعلية باستخدام آردوينو


المؤسسة العامة للتعليم الفني والتدريب المهني


GEEKS
valley

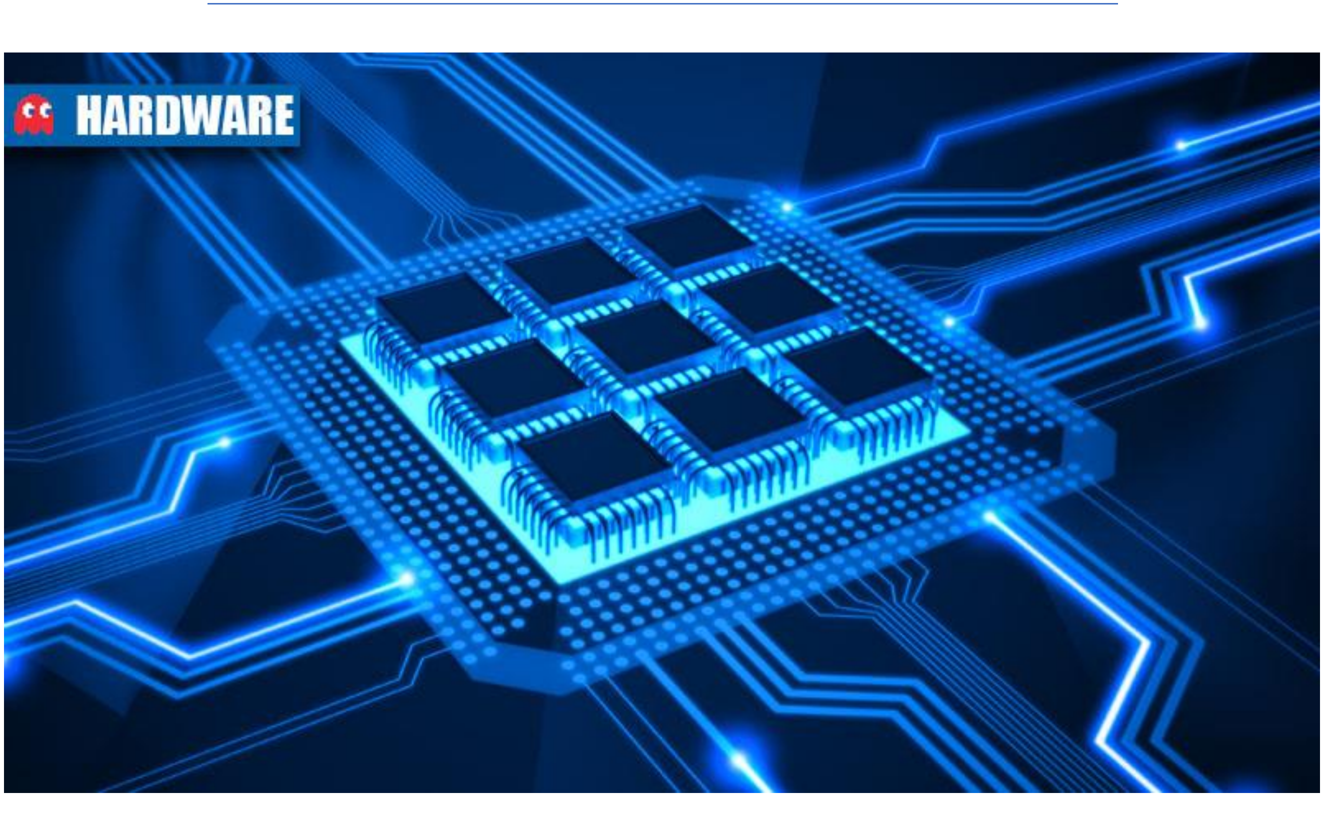
## جدول المحتويات


| العنوان | رقم الصفحة |
| --- | --- |
| رخصة الكتاب | 4 |
| عن الكاتب | 5 |
| اهداء السلسلة | 6 |
| المقدمة | 7 |
| جدول المحتويات | 15 |
| تطور المتحكمات الالكترونية | 18 |
| الاشارات التماثلية والرقمية | 41 |
| تعديل عرض النبضة | 49 |
| انظمة العد | 55 |
| لوح التوصيل Breadboard | 69 |
| الدوائر المتكاملة | 89 |
| الدوائر المنطقية | 152 |
| الميكرو كونترولر | 225 |
| انواع الميكروكونترولر | 228 |
| مُبرمجة الميكرو كونترولر | 247 |
| تجهيز بيئة التطوير | 271 |
| برمجة الميكروكنترولر | 282 |

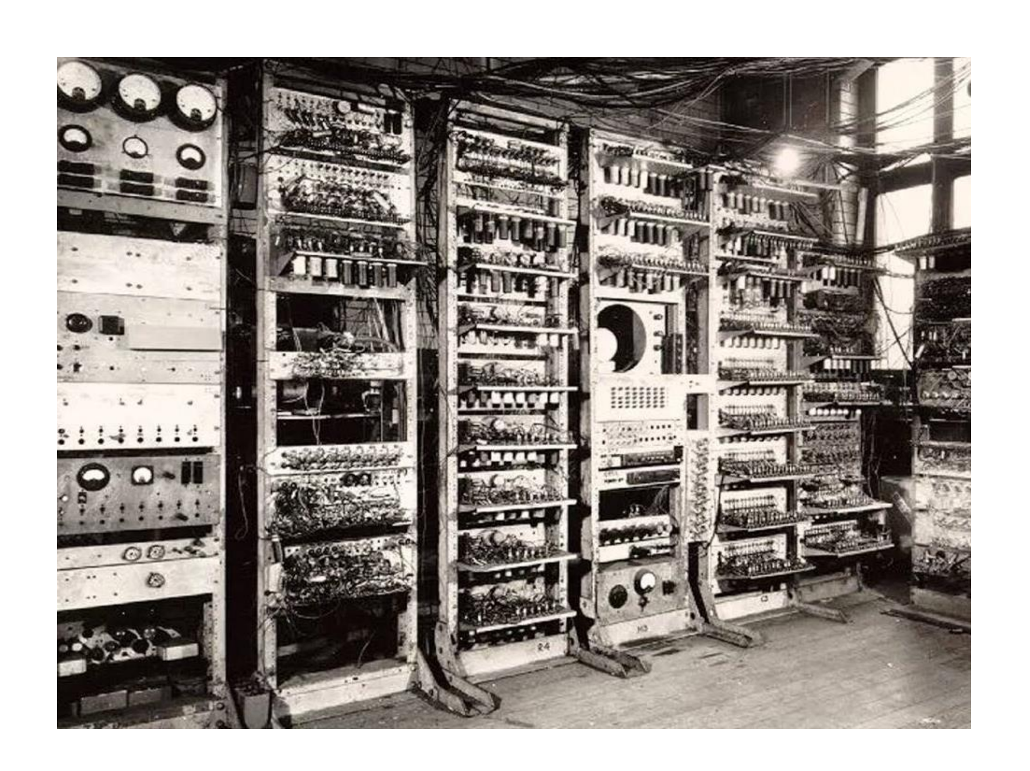
| | |
|---|---|
| 309 | الشرط والدوران |
| 318 | المصفوفات والدوال |
| 341 | كيفية قراءة المُخططات |
| 365 | الاردوينو |
| 368 | انواع الاردوينو |
| 375 | تجهيزات ادوات الاردوينو الاساسية |
| 378 | تجهيز بيئة التطويرIDE |
| 386 | تهيئة الاردوينو |
| 390 | برمجة الاردوينو |
| 407 | المصابيح LED |
| 420 | المكتبات |
| 422 | استخدام PWM |
| 428 | المداخل التماثلية في الآردوينو |
| 433 | استخدام الحساسات |
| 440 | المنفذ التسلسلي |
| 470 | عرض بيانات الحساسات |
| 478 | شاشات العرض |
| 492 | المحركات |
| 543 | التحويل من تناظري إلى رقمي |
| 548 | التحكم عن بعد |

| | |
|---|---|
| لوحة المفاتيح | 637 |
| المرحلات | 658 |
| المشغلات | 666 |
| الاغطية | 671 |
| برمجة الاردوينو باستخدام الهاتف | 682 |
| مقدمة في الراسبيري باي | 686 |
| انواع الراسبيري باي | 691 |
| اردوينو ام راسبيري باي ؟ | 696 |
| تهيئة الراسبيري باي | 706 |
| برمجة الراسبيري باي باستخدام البايثون | 717 |
| وحدة التحكم المنطقية القابلة للبرمجة (PLC) | 725 |
| ما هو ال PLC | 725 |
| مكونات ال PLC | 728 |
| انواع واشكال ال PLC | اصدار تجريبي |
| برمجة ال PLC | اصدار تجريبي |
| بناء تطبيقات صناعية باستخدام ال PLC | اصدار تجريبي |
| الملحق | 735 |
| المراجع والمصادر | - |

## تطور المتحكمات الالكترونية



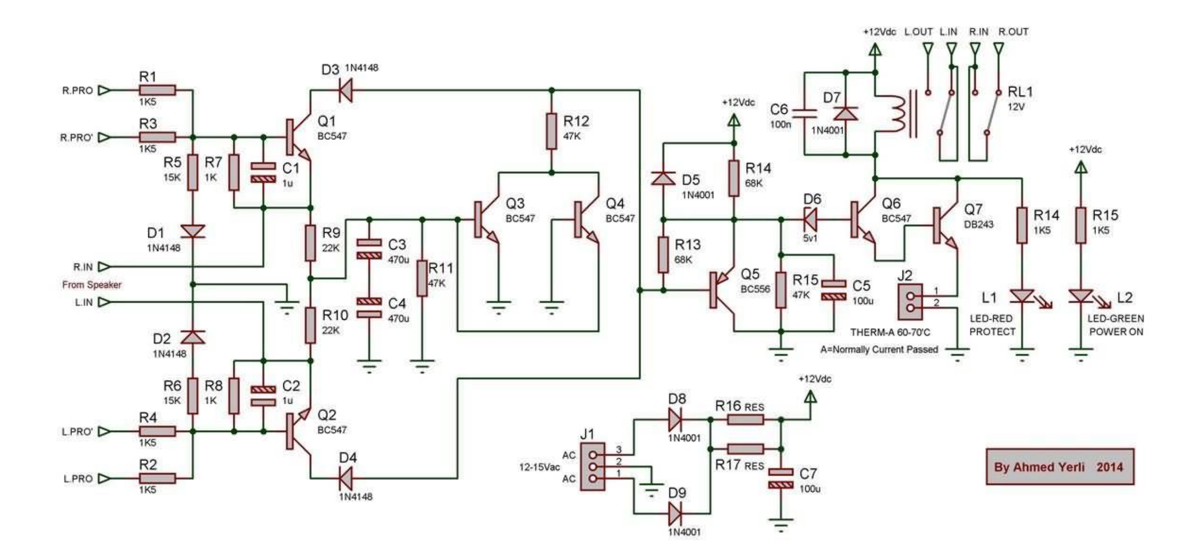

منذ زمن ليس ببعيد كانت الصناعات الالكترونية في غاية من التعقيد بسبب كثرة استخدام العناصر الالكترونية فكانت المشاكل كبيرة مثل الضجيج والامواج المبعثرة وارتفاع درجات الحرارة مما يؤدي الى ارتفاع نسبة الخطورة , فقديما كان للعمل على صناعة دائرة الكترونية يلزم المشروع الكثير من العناصر الالكترونية مثل المقاومات,المكثفات , الترانزستورات , المحولات ... الخ . وهذا كان من شأنه زيادة التعقيد فيمكنك تخيل تعقيد الامر من خلال الصورة التالية التي توضح اول جهاز حاسوب في العالم عام 1941.



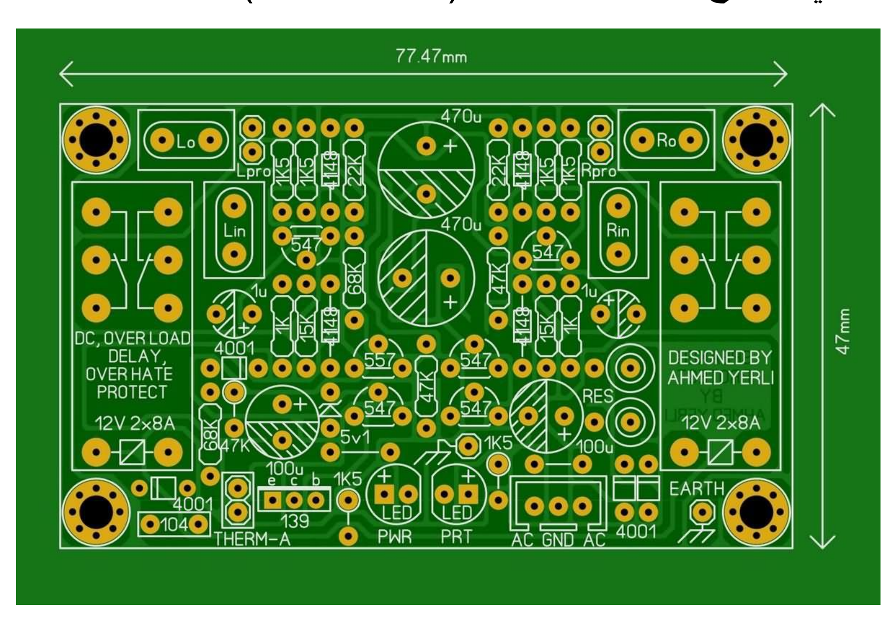
كان الحاسوب ضخم بحجم غرفة كبيرة ويستهلك كثيرا من الطاقة الكهربائية ولا يقوم الا بوظائف ومهام قليلة جدا واعادة تغيير او تعديل جزء بسيط من اي دائرة الكترونية كان يعني الكثير من العمليات المعقدة من اللحام وقطع الاسلاك واعادة النظر في مخططات كهربائية كبيرة ! والكثير من الامور المزعجة والتي ادت الى اقتصار وظيفة تطوير المنتجات الالكترونية على مجموعة مختصين من المهندسين فقط.

مع تطور التكنولوجيا والاجهزة الالكترونية في الاستخدامات المنزلية والصناعية والتجارية وادارة المصانع , اصبح من الضروري ايجاد حلول لضخامة حجم الدوائر الالكترونية ومن هنا جائت الحاجة لاختراعات جديدة تغير التقنيات الى افضل حال وفي هذا الكتاب سوف نوضح هذه الاختراعات التي غيرت مجرى العالم !

المخطط التالي يوضح جهاز (SPEAKER PROTECTION CIRCUIT)




والمخطط التالي يوضح الدائرة المطبوعة (لوحة التوصيل) والعناصر الالكترونية





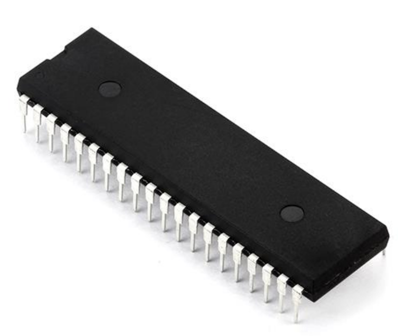
يبدو كبيرا اليس كذلك ؟ يحتاج الى الكثير من العناصر الالكترونية واداوت ومواد اللحام والكثير من الثقوب ومن الصعب اكتشاف الاعطال بسهولة كل هذا فقط لتمثيل دائرة صغيرة من دوائر من جهاز التلفاز! , لذلك من هنا تكمن اهمية المتحكمات الالكترونية فعن طريق متحكم مثل الموجود في الصورة المجاورة يمكن الاستغناء عن المخطط الكبير ومن الجدير ذكره ان هذه القطعة اكثر كفاءة واستقرار واقل كلفة وسهولة في الفك والتركيب .

لنبدأ رحلة التعرف على اهم المتحكمات الالكترونية , في الحقيقة ان المتحكمات الالكترونية تقسم الى نوعين اساسين :

- متحكمات الكترونية غير قابلة للبرمجة : وهي النوع الذي يضم عناصر ومكونات الكترونية داخل شريحة صغيرة .
- متحكمات الكترونية قابلة للبرمجة : هذا النوع من المتحكمات يحتوي على مكونات تشبه تلك المكونات الموجودة في اي جهاز حاسوب فهي تحتوي على وحدة تخزين وذاكرة وصول عشوائية والكثير من التفاصيل التي سنتعرف عليها , لكن في هذا الكتاب سنركز على متحكم الكتروني يسمى الاردوينو لنكمل به دروس هذه السلسلة.

سنوضح انواع المتحكمات الالكترونية وتفاصيلها حسب التسلسل الزمني لها:

- الدارات المتكاملة Integrated Circuit



كانت أول فكرة للدارات المتكاملة، قد بدأت على يد عالم رادار بريطاني يدعى جيفري دُمِر ، مواليد 1909. والذي كان يعمل لدى المؤسسة الملكية للرادار والتابعة لوزارة الدفاع البريطانية. وأعلن عنها في واشنطن في أيار 1952. ولكنه لم يتمكن من تصنيعها أبداً. الفكرة الأساسية للدارات المتكاملة هي إنشاء مربعات خزفية ، كل منها يحتوي عنصراً منمنماً (بالغ الصغر). وهذه العناصر يمكن أن تدمج فيما بعد وتوصل إلى شبكة مدمجة ثنائية الأبعاد أو ثلاثية. هذه الفكرة والتي بدت واعدة جداً عام 1957 قدمت للجيش الأمريكي من قبل جاك كيلبي، والذي قدمت له كافة الحوافز الممكنة، ليقدم أخيراً تصميماً ثورياً جديداً هو الدارات المتكاملة والتي أصبحت تعرف حاليا IC أول دارة متكاملة صنعت بشكل مستقل من قبل عالمين هما: جاك كيلبي، العامل لدى شركة Texas Instruments، والتي كانت عبارة عن "دارة صلبة" مصنوعة من الجرمانيوم، والعالم الآخر هو روبرت نويس"Robert Noyce" ، والذي كان يعمل في شركة Fairchild Semiconductor والذي قام بصنع دارة أكثر تعقيداً من سابقتها وأساسها السيليكون.

هكذا ولدت الدارات المتكاملة، ومن حينها بدأت عمالقة شركات صناعة الإلكترونيات منافستها على إنتاج أصغرها، أسرعها، وأفضلها أداءً. حتى أصبحت موجودة في كل شيء كهربائي تقريباً. والدارة المتكاملة عبارة عن دارة بكاملها موجودة في قطعة صغيرة من السيليكون أبعادها مختلفة فيمكن ان تكون (1.5 mm x 1.5 mm x 0.2 mm) ، وتحتوي على عدد كبير من العناصر الإلكترونية قد تصل للالاف : ترانزستورات، متصلات ثنائية، مقاومات، مكثفات . وذلك حسب نوعها. وتتصل الدارة المتكاملة مع الدارة الخارجية بواسطة ما يدعى دبابيس (Pins).

ويمكننا تعريف ال IC كالتالي :

تختصر إلى (IC) أو الشريحة الإلكترونية (Chip) عبارة عن دائرة الكترونية مصغرة وهي من ضمن مايعرف بتقنية ميكروية والتي هي بدورها جزء من الهندسة الإلكترونية، أحدث ثورة في عالم الإلكترونيات, الشريحة رقيقة من مادة السيلكون تبلغ مساحتها عدة ملليمترات ويطلق عليها ((شريحة السيلكون)) أو رقاقة السيلكون وتحتوي

شريحة السيلكون على الآلاف من المكونات الإلكترونية الدقيقة جدا، مثل الترانزستورات والمقاومات والمكثفات التي تربط معا لتكون دوائر إلكترونية متكاملة، والشرائح الإلكترونية التي تستخدم في الوقت الحاضر، تحتوي على عشرات الألوف من المكونات المختلفة، التي تحشر في مساحة تبلغ حوالي 30 - 40 ملليمتر مربع، ويمكنها أن تخزن 64,000 وحدة من المعلومات (بيت bit). ويمكن حاليا إنتاج المئات من هذه الرقائق دفعة واحدة، وذلك باستخدام وسائل تقنية حديثة مختلفة تشتمل على عدة عمليات دقيقة متتابعة، مثل عمليات التصوير باستخدام قوالب معينة، والحفر، وترسيب مواد كيميائية تعمل كشوائب (مثل ذرات الفسفور) داخل الشبكة البلورية لعنصر السيليكون، وذلك لتصبح الشريحة السيليكونية الواحدة في النهاية تعمل عمل أشباه الموصلات semiconductors والتي تبنى منها الترانزستورات. وفي ختام عملية إنتاج الشرائح الإلكترونية، يتم وضع طبقة من مادة الألمنيوم يتكون منها أحجبة ثم تحفر لتكوين الروابط بالدوائر الخارجية.

المميزات:

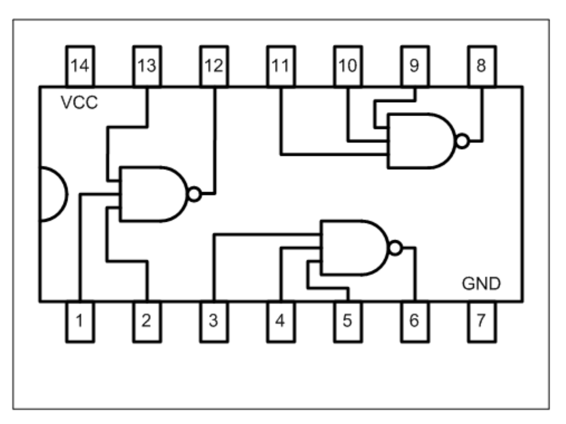
- حجمها صغير ويمكن أن يصل 10/1 بوصة مربعة.
- استهلاكها للقدرة الكهربائية ضعيف مقارنة بالأنواع الأخرى من العناصر.
- تكلفة أقل (رخيصة الثمن).
- تعمل الدائرة المتكاملة بكفاءة عالية ربما تصل إلى 50 مرة من كفاءة الدوائر العادية.
- تعمل بسرعة عالية حيث أن الإشارة تأخذ زمناً أقل عند انتقالها داخل الدائرة.
- عدم وجود لحامات داخلية يقلل من احتمال حدوث فصل داخلي للأطراف حيث أن المكونات تتصل ببعضها عن طريق شرائح رقيقة من المعدن.

العيوب :

- لا تعمل بتيارات عالية بسبب صغر الحجم.
- تتأثر الدوائر المتكاملة بدرجة الحرارة ولذلك فهي تحتاج إلى وسيلة للتبريد عند العمل على قدرات عالية.

- لا يمكن تصنيع بعض المكونات داخل الدائرة المتكاملة مثل الملفات نظراً لكبر حجم الملف المصنع بإستخدام طريقة تصنيع الدوائر المتكاملة وكذلك المكثفات ذات السعات الكبيرة.
- لا يمكن إصلاح الدوائر المتكاملة عند تلفها ولكن يتم إستبدالها.

▪ الدوائر المنطقية Logical Circuits




هي دائرة إلكترونية تحتوي على (مدخل واحد أو عدة مداخل) ومخرج واحد حيث تقوم بعملية منطقية على المدخل وتنتج المخرج المطلوب، تستخدم هذه البوابات في بناء معالجات الأجهزة الاكترونية والحواسيب. لأنّ مُخرج البوابة الرقمية هو أيضاً قيمة منطقية، فإنّه يمكن استخدام مخرج أحد البوابات المنطقية كمدخل لبوابة أخرى. المنطق المستخدم غالباً هو المنطق البولياني ، وهو المنطق الذي يعمل في الدوائر الرقمية.

يتم صناعة الدائرة الإلكترونية للبوابة الرقمية باستخدام دايودات ، ويمكن أيضاً بناؤها باستخدام المبدّلات الإلكترونية، سوائل منطقية، إشارات ضوئية، جزيئات، وحتى من أجزاء ميكانيكية.

هناك ثلاثة بوابات أساسية سوف ندرسها وهم :

- البوابة AND
- البوابة OR
- البوابة NOT

- الميكرو كونترولرMicrocontroller (المتحكم الدقيق)



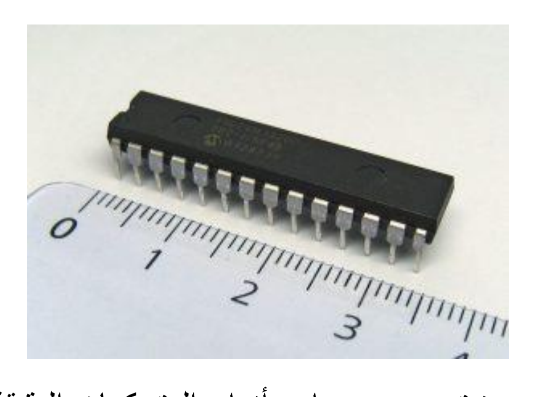

لا تخلو حياتنا اليومية من استخدام التكنولوجيا ووجود المتحكمات الدقيقة في كل مكان حولنا أصبح أمراً لابد منه , المتحكمات الدقيقة microcontrollers هي عبارة عن كمبيوترات صغيرة على شريحة قابلة للبرمجة لتؤدي مجموعة من المهام يحتوي على وحدة معالجة مركزية CPU، ذواكر Memories، وواجهات اتصال Interfaces.
الفرق الأساسي بين الكمبيوترات والمتحكمات الدقيقة هي المهام التي تؤديها، مثلاً تستخدم الكمبيوترات لتشغيل الأفلام ومشاهدتها او لتصفح الانترنت او قراءة الكتب والمقالات . بينما تستخدم المتحكمات الدقيقة في المنتجات او الاجهزة التي يتم التحكم فيها اتوماتيكياً وتوجد الكثير من المنتجات والاجهزة التي تجعل استخدامنا لها يومياً بدون ان ندرك ذلك مثل التلفزيون، الجوالات، أنظمة الأمن، الكاميرات، أجهزة ألعاب الفيديو، المايكرويف، الطابعات، السيارة ويمكن تفصيل بعضها كالآتي:

- المكيفات للتحكم بدرجة الحرارة أو مدة التشغيل
- الغسالات للتحكم بسرعة واتجاه المحركات والمهام التي تقوم بها (غسل، شطف، تنشيف)
- التحكم بالاضاءة حيث يمكن او تعمل حين يواجد شخص ما في الغرفة او أن تعمل في وقت محدد

- محطات الطقس كقراءة درجة الحرارة والرطوبة ومستوى أشعة الشمس وسرعة الرياح ومستويات الغازات, ويمكن حفظها وعمل إحصاءات او مقارنات بينها.
- التحكم في الروبوتات: فمثلا التحكم في سرعته، مساره، حركة الأذرع ، قراءة المعلومات (صوت أو فيديو)

- يمكن رؤية المتحكمات الصغرية في كثير من الأجهزة الإلكترونية بدء من الألعاب الصغيرة وحتى المصانع المؤتمتة ، فهي تسيطر على معظم سوق تطبيقات المعالجات. أكثر من 50% من المتحكمات الصغرية من النوع "البسيط" و حوالي 20% منها عبارة عن معالجات إشارات رقمية عالية التعقيد(DSPs). بعض السيارات تحتوي على مايزيد عن 50 وحدة من هذه المتحكمات.

يمكننا القول انه عبارة عن دارة الكترونية متكاملة تحتوي على معالج دقيق داخلي وذاكرة داخلية قابلة للبرمجة لتخزين البرنامج التحكمي فيها وذاكرة أخرى لمعالجة البيانات كما أنها تحتوي على بوابات إدخال وإخراج البيانات والأوامر التحكمية كما وقد تحتوي على أدوات أخرى كالمبدل التماثلي الرقمي وبالعكس وعلى مقارنات الجهد ومكبرات العمليات و مولد نبضات الساعة والعدادات والمؤقتات وغيرها .




صورة توضح حجم احد أنواع المتحكمات الدقيقة

نقوم يومياً بالتعامل مع المتحكمات الدقيقة في الأجهزة الكهربائية والالكترونية ولكن معظمنا يتعامل معها كمشغل أو كمستخدم، والتعامل معها كمبرمج او متحكم سهل جداً ولا يحتاج إلى الكثير من المعرفة او الخبرة الهندسية في البرمجة والالكترونيات ودوائرها، ودليل ذلك هو وجود كثير من الهواة غير المتخصصين في هذا المجال حول العالم.

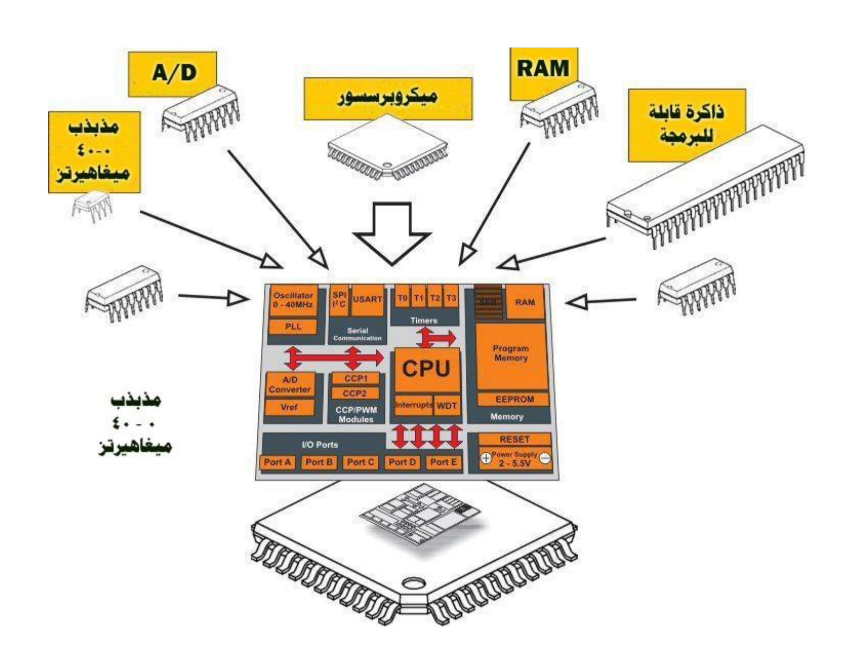
بعض أنواع اللوحات التي تحتوي المتحكمات الدقيقة بداخلها :

- الاردوينو Arduino

هناك فئتين أساسيتين للمعالج التحكمي من الناحية الوظيفية الفئة الأولى هي فئة الأغراض العامة وهي الأنواع التي يمكن برمجتها للقيام بأي نوع من المهمات طالما كان في نطاق قدرتها من حيث الإمكانية وليس النوعية كالمتحكمات العامة التي يمكن استخدامها في جهاز الانذار أو بطاقة الربط أو التحكم بالإنارة وهناك النوع الثاني وهي المتحكمات ذات الوظائف الخاصة والتي بنيت لتنجز مهام من نوع مخصص مع إمكانية برمجتها للتحكم بسر العمل في إنجاز هذه المهام مثل المتحكم الخاص بفك تشفير جهاز استقبال الانترنت مثلا.

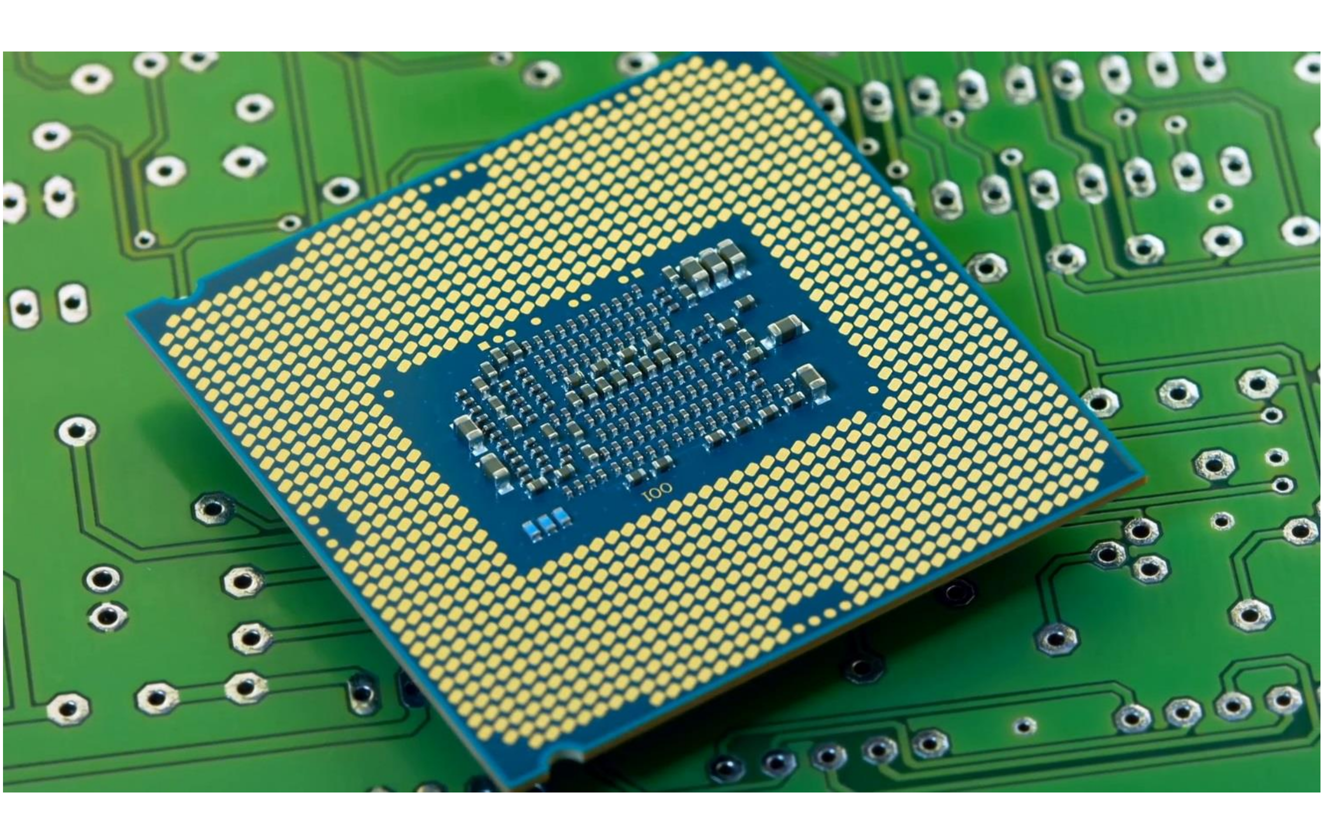
❖ المكونات الاساسية للمتحكمات الدقيقة

تعمل المتحكمات الصغرية في الغالب وفق معمارية هارفرد Harvard Architecture فيما يلي أبرز مكونات المتحكم الصغري:



- وحدة معالجة مركزية CPU وتتراوح ما بين 8 إلى 32 وحتى إلى 64 خانة bit
- واجهة المدخلاتI/O Interfaces
- الملحقات Peripherals كالمؤقتات, الراصدwatchdog
- ذواكر الوصول العشوائي لتخزين البياناتRAM for Data Storage
- ذواكر كتابة فقط ROM ، Flash memory , EEPROM
- مزامن clk Generator

- الميكرو بروسسير Microprocessor (المعالج الدقيق)



مهما كان الجهاز الذي تستخدمه من أجل قراءة هذا الكتاب ، فإنه يتضمن بداخله وحدةً أساسية تعتبر النواة والدماغ المحرك لكل أجزائه ووحداته، وهذه الوحدة هي المعالج الصغري MPU: Microprocessor فسواء كنت تستخدم حاسوب مكتبي، أو حاسوب محمول، أو حتى هاتفاً ذكياً أو جهازاً لوحياً، فإن المعالج الصغري المتواجد داخل أحد هذه الأجهزة هو القطعة الأساسية التي تكونه وتقود فعالياته.

وكي يكون المفهوم وأكثر شمولية فالمعالج الصغري هو الأداة التي تقوم بقيادة وتنظيم عمل أي نظام حاسوبي، سواء كان هذا النظام عبارة عن حاسوب مكتبي، حاسوب محمول، خادم شبكة Server ، أو حتى حاسوب فائق SuperComputer

يعرف أيضاً المعالج الصغري بـ وحدة المعالجة المركزية CPU: Central Processing Unit والتي تتواجد بقلب أي نظام حاسوبي وتشكل نواته الأساسية،

وهو عبارة عن دارة متكاملة قابلة للبرمجة ، يتم تصميمها وفقاً لمعمارية محددة، من أجل تنفيذ عمليات معالجة البيانات، والحصول على المعلومات اللازمة من أجل قيادة النظام الحاسوبي ككل. وتعتبر المعالجات الصغرية نتيجةً لتطور تقنية الدارات المتكاملة وأحد أهم تطبيقات ثورة أنصاف النواقل التي تمت بمجال الصناعة التقنية بعد أن قام وليام شوكلي باختراع وتطوير الترانزستور عام 1947. فبعد تقديم الترانزستور، أصبح العلماء والتقنيين قادرين على تنفيذ مهامٍ متنوعة باستخدام الترانزستورات، وأهمها القدرة على تمثيل المنطق الرقمي الذي يتعامل مع البيانات على أنها وحدات وأصفار، وذلك عبر استغلال ميزة سرعة الفتح والإغلاق التي يتمتع بها الترانزستور، فضلاً عن حجمه الصغير (مقارنةً بالصمامات الكهربائية المفرغة المستخدمة سابقاً). ومع تطور تقنيات التعامل مع العناصر الإلكترونية نصف الناقلة (أو شبه الموصلة)، تطوت تقنية الدارات المتكاملة، والتي تعني القدرة على وضع العديد من العناصر نصف الناقلة ضمن شريحةٍ إلكترونية واحدة. ظهور تقنية الدارات المتكاملة سمحت بتطوير الدارات الإلكترونية بشكلٍ كبير، كونها تسمح بدمج العديد من الترانزستورات ضمن الشريحة الواحدة، فضلاً عن إمكانية تشكيل هذه الترانزستورات ضمن الشريحة وفقاً لهيكليات مختلفة. هذه التطورات كلها قادت لظهور المعالجات الصغرية في بداية سبعينيات القرن الماضي.

ففي عام 1971، أعلنت شركة إنتل عن تطوير أول معالج صغري، وهو المعالج الصغري 4004، الذي كان يتمتع بسعة نواقل قدرها 4 بت. لم يكن المعالج 4004 معالجاً فعالاً وذو قدراتٍ فعالة في معالجة البيانات وتنفيذ العمليات الحاسوبية المختلفة، إلا أن التمكن من تصميمه وتنفيذه فتح الباب أمام تطويره للحصول على معالجاتٍ أكثر كفاءة. وقد استخدم المعالج الصغري 4004 من أجل قيادة عمل بعض الآلات الحاسبة الإلكترونية البسيطة، والتي كانت سابقاً تعتمد على تشكيلات متنوعة من العناصر الإلكترونية المتوضعة بشكلٍ منفصل، من أجل إتمام المهام الحسابية المطلوبة منها.

ولكن ماذا حصل بعد إنتاج المعالج الصغري 4004؟

بعد إنتاج أول معالج صغري من قبل شركة إنتل عام 1971، تبين أن المعالج غير كفؤ وغير مناسب من قيادة نظام حاسوبي متكامل. وبعد ذلك، عملت الشركة على تطوير معالجها الجديد عام 1974: المعالج الصغري 8080، والذي يعتبر أول معالج صغري تم تضمينه ضمن حاسوب، وهو يمتلك سعة خطوط مقدارها 8-بت. وبعد المعالج

8080، تم تطوير المعالج 8085 بسعة خطوط عنونة أيضاً 8-بت، ثم المعالج الصغري 8086، ثم تم تطوير المعالج الصغري 8088 الذي يعتبر أول معالج صغري تم تحقيق نجاح كبير في السوق التجارية. تم إنتاج المعالج الصغري 8088 عام 1979، وقد تم تضمينه ليكون نواة المعالجة الأساسية في حواسيب شركة IBM التي ظهرت عام 1982.




المعالج 4004 من شركة إنتل هو أول معالج صغري تم إنتاجه على الإطلاق. كان هذا المعالج قادر على تشغيل آلة حسابية بسيطة.

وبعد ظهور المعالج 8088، استمر تطور المعالجات الصغرية من قبل شركة إنتل، فظهر المعالجات التالية خلال فترتي الثمانينيات والتسعينيات: 80286 وهو أول معالج صغري بسعة خطوط عنونة مقدارها 16-بت، ثم المعالج 80386 الذي كان أول معالج صغري بسعة خطوط عنونة مقدارها 32-بت، ثم المعالج 80486، وأخيراً المعالج 80586 الذي عرف باسمه الشهير ."Pentium 1" وبعد المعالج Pentium 1 استمرت مسيرة التطور وصولاً إلى المعالج المتفوق الشهير Pentium 4 عام 2000.

ثم ظهر المعالج Itanium عام 2001 ليكون أول معالج صغري بسعة خطوط عنونة مقدارها 64-بت، ثم بدأت أجيال المعالجات متعددة النواة بالظهور: المعالج ثنائي النواة Core 2 Duo في عام 2006، والمعالج رباعي النواة Core 2 Quad في عام 2007، والمعالج المتفوق Core 2 Extreme,Quad Processor في عام 2008، ثم ظهرت معالجات الجيل الرابع. Core i3, Core i5, Core i7 :

وكي نسلط الضوء أكثر على تطور المعالجات بشكل واضح أكثر، فإنه ينبغي على القارئ أن يعرف أن أول معالج صغري 4004 كان يضم 2300 ترانزستور على الشريحة، وكانت يعمل عند تردد 108 كيلو هرتز، بينما المعالج Core 2 Extreme قد وصل عدد الترانزستورات على الشريحة إلى 820 مليون ترانزستور على الشريحة، ويعمل عند تردد 3.2 غيغا هرتز، بينما معالجات الجيل الرابع فقد تجاوز فيها عدد الترانزستورات على الشريحة المليار ترانزستور (1.4 مليار ترانزستور على الشريحة بالنسبة للمعالج Core i7). وبالنسبة لعائلة معالجات Core M التي أطلقتها شركة إنتل في عام 2014، فإن الشركة ستقوم بتصنيع هذه المعالجات اعتماداً على ترانزستورات بأبعاد من رتبة 14 نانومتر، لتكون أول شركة في العالم تستطيع كسر هذا الحاجز والوصول لهذه الدرجة العالية من التكامل.

## تطور معالجات شركة إنتل ومواصفاتها الأساسية

| اسم المعالج | تاريخ الصنع | عدد الترانزستورات | طول أصغر ناقل ضمن الشريحة (ميكرون) | تردد ساعة المعالج Clock Speed | عرض البيانات Data Width | ملايين العمليات في الثانية MIPS |
|---|---|---|---|---|---|---|
| 8080 | 1974 | 6000 | 6 | 2 ميغا هرتز | 8-بت | 0.64 |
| 8088 | 1979 | 29000 | 3 | 5 ميغا هرتز | 8-بت | 0.33 |
| 80286 | 1982 | 134000 | 1.5 | 6 ميغا هرتز | 16 بت | 1 |
| 80386 | 1985 | 275000 | 1.5 | 16 ميغا هرتز | 32-بت | 5 |
| 80486 | 1989 | 1200000 | 1 | 25 ميغا هرتز | 32-بت | 20 |
| Pentium | 1993 | 3100000 | 0.8 | 60 ميغا هرتز | 32-بت | 100 |
| Pentium2 | 1997 | 7500000 | 0.35 | 233 ميغا هرتز | 32-بت | 300 |
| Pentium3 | 1999 | 9500000 | 0.25 | 450 ميغا هرتز | 32-بت | 510 |
| Pentium4 | 2000 | 42000000 | 0.18 | 1.5 غيغا هرتز | 32-بت | 1700 |
| Pentium4 "Prescott" | 2004 | 125000000 | 0.09 | 3.6 غيغا هرتز | 32-بت | 7000 |

الخط الزمني لتطور معالجات شركة إنتل مع أهم المواصفات الأساسية المتعلقة بالمعالجات، بدءاً من المعالج 8080 وحتى المعالج Pentium4.

حسناً، قد تشعر هنا بالقليل من التشتت والضياع فكيف سيستطيع الشخص أن يستوعب عمل كافة هذه المعالجات الصغرية، وقد تطور عملها وأدائها بشكلٍ كبير؟ وما الذي يجب أن يفهمه الشخص بالضبط؟

في الواقع، فإن عرضنا السابق لتطور المعالجات الصغرية بدءاً من المعالج 4004 لم يكن بغرض التشتيت، وإنما بهدف تسليط الضوء على التطور الكبير الذي حصل بمجال

صناعة المعالجات الحاسوبية. والأهم من ذلك، فإننا نود أن نشرح بعض المواصفات الأساسية التي تتعلق بأداء المعالجات الصغرية , وهي كما يلي:

1 -تاريخ الصنع: هو العام الذي ظهر فيه المعالج لأول مرة. هنالك العديد من المعالجات التي تم إعادة تقديمها، وذلك بعد إضافة بعض التحسينات إليها.

2 -عدد الترانزستورات: يشير إلى عدد الترانزستورات المتوضعة على شريحة المعالج. ويمكن ملاحظة أن عدد الترانزستورات على الشريحة قد تضاعف بشكلٍ كبير مع مرور الزمن (قانون مور)

3 -طول أصغر ناقل ضمن الشريحة :Micron تشير هذه الخاصية إلى طول أصغر سلك ناقل ضمن شريحة المعالج، مقدراً بواحدة الميكرون، والميكرون أصغر من السنتمي متر بـ 1000 مرة، وشعرة الإنسان – على سبيل المثال – تمتلك ثخانةً قدرها 100 ميكرون.

4 -تردد عمل المعالج Clock Speed تشير هذه الخاصية إلى أقصى معدل سرعة يمكن للشريحة أن تعمل وفقه. سيتم تسليط الضوء على هذا المفهوم بشكلٍ أكبر في الفقرات المقبلة.

5 -عرض البيانات Data Width أو سعة خطوط العنونة. هذه الخاصية تتعلق بشكلٍ أساسي بوحدة الحساب والمنطق ALU ضمن بنية المعالج الصغري، وهي تشير إلى كمية البيانات التي تستطيع النواقل الموجودة ضمن بنية المعالج بنقلها و/أو معالجتها بتعليمةٍ واحدة. هذا يعني أن المعالج الذي يمتلك سعة عرض بيانات قدرها 8-بت، يستطيع أن يقوم بإجراء عمليات الجمع، الطرح، الضرب، القسمة على أرقامٍ بطول 8-بت بكل تعليمة معالجة. وفي حال كانت الأرقام بطول 32-بت – مثلاً – فإن معالج بعرض خطوط عنونة 8-بت سيحتاج لـ 4 تعليمات من أجل تنفيذ العمليات الحسابية المطلوبة على الأرقام التي تمتلك طول قدره 32-بت، بينما يستطيع معالج بعرض خطوط عنونة 32-بت أن يقوم بالعمليات الحسابية اللازمة على هذه الأرقام بتعليمةٍ واحدة.

6 -ملايين العمليات في الثانية MIPS وهو أحد معايير قياس كفاءة وقدرة المعالج، وهو يشير إلى عدد العمليات التي يستطيع المعالج أن يقوم بتنفيذها في ثانيةٍ واحدة، ولكن مقدرةً بالملايين، والمصطلح MIPS هو اختصار لـ Millions of Instructions per Second

ومن الجدول السابق، يمكن أن نلاحظ أن هنالك علاقة ما بين تردد عمل المعالج، وما بينMIPS ، فكلما كانت تردد عمل المعالج أسرع، كلما كان عدد التعليمات المنفذة في ثانيةٍ واحدة أكبر. كما أنه يوجد هنالك علاقة ما بين زيادة عدد الترانزستورات على شريحة المعالج، وما بينMIPS

منطق المعالجات الصغريةMicroprocessors Logic

من أجل فهم كيفية المعالج الصغري، فإنه من المفيد النظر بداخله ومعرفة ما هو المنطق الذي يحكم عمل المعالجات الصغرية، كما أنه من المفيد أيضاً معرفة المعلومات الأساسية المتعلقة بلغة التجميعAssembly Language ، والعديد من الأمور والتفاصيل الأخرى التي يستطيع المهندسون استخدامها من أجل فهم آلية عمل المعالجات الصغرية وكيفية تعزيز قدرتها وسرعتها.

بشكلٍ أساسي، فإن المعالج الصغري يقوم بتنفيذ مجموعة من تعليمات لغة الآلة Machine Instructions، حيث تحدد هذه التعليمات كيفية قيام المعالج لعمله، واعتماداً على هذه التعليمات، فإن المعالج يقوم بالمهام التالية:

يقوم المعالج الصغري بعملياتٍ حسابية مثل الجمع، الطرح، الضرب، والقسمة، اعتماداً على وحدة الحساب والمنطق الموجودة داخله .ALU المعالجات الصغرية الحديثة تتضمن معالجاتٍ ووحداتٍ خاصة من أجل تنفيذ العمليات الحسابية ، حيث تستطيع أن تقوم بتنفيذ عملياتٍ حسابية معقدة ومتنوعة على هذه الأعداد.

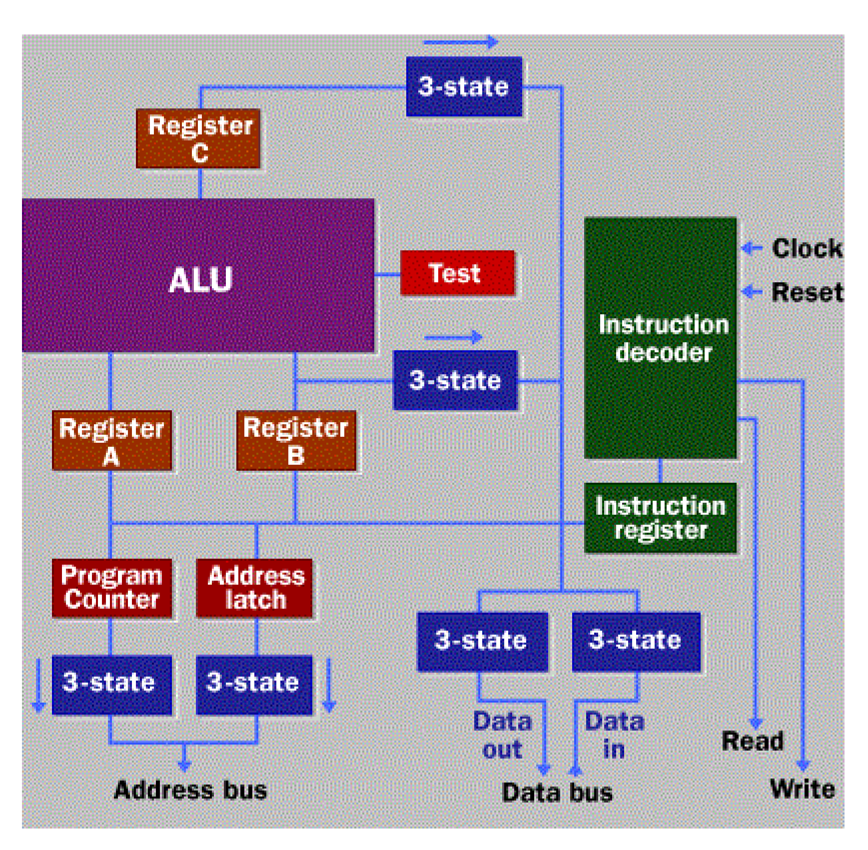
يقوم المعالج الصغري بنقل البيانات من أحد مواقع الذاكرة إلى موقعٍ آخر ويستطيع المعالج الصغري أن يقوم بعمليات اتخاذ القرار، والانتقال إلى مجموعةٍ جديدة من التعليمات بناءً على القرارات التي اتخذها واستخدام الذكاء الاصطناعي .

بالطبع، فإن المعالج الصغري يقوم بالعديد من المهام المعقدة، ولكن بشكلٍ أساسي، فإن المهام السابقة هي المهام الأساسية له. ومن أجل توضيح المهام المختلفة التي يقوم بها المعالج الصغري، فإننا سنوضحها بالمخطط التالي:

ناقل العناوين Address Bus: هو الخط الناقل الذي يقوم بإرسال العناوين إلى الذواكر.
ناقل البيانات Data Buss: هو الخط الناقل الذي يقوم بإرسال البيانات إلى الذاوكر، أو استقبال البيانات من الذواكر.
خطوط القراءة والكتابة Read/Write Line: وهي التي تقوم بإخبار الذواكر فيما إذا كانت تريد أن تقوم بالحصول على عناوين لمواقع معينة، أو تريد أن تقوم بضبط عناوين هذه الموقع.
خط الساعة Clock Line والذي يسمح لنبضات الساعة أن تقوم بتنظيم تردد عمل المعالج.
خط التصفير Reset Line: يقوم بتصفير القيمة الموجودة في عداد البرامج Program Counterإلى الصفر (أو أي قيمة أخرى) ويقوم بإعادة تشغيل عملية التنفيذ.
لنفترض الآن أن ناقل العناوين وناقل البيانات يمتلكان عرضاً قدره 8-بت، فإن مكونات المعالج الصغري الموافق لهما ستكون كما يلي:

السجلاتA ، B، Cهي عبارة عن ماسكات Latches مصنوعة من دارات القلابات Flip-Flop، وماسك العناوين هو عبارة عن سجل مشابه تماماً للسجلات.A, B, C
عداد البرامج Program Counter هو أيضاً عبارة عن دارة ماسك، مع ميزةٍ إضافية هي قدرته على زيادة القيمة المحفوظة ضمنه بمقدار واحد، وذلك عندما يتم إخباره بذلك، كما يمتلك القدرة على تصفير القيمة المخزنة ضمنه، أيضاً عندما يتم إخباره بذلك.
وحدة الحساب والمنطق ALU يمكن أن تكون ببساطة عبارة عن دارة جامع منطق بعرض 8-بت، أو قد تكون دارة قادرة على أداء عمليت الجمع، الطرح، الضرب، القسمة، وذلك لقيم بعرض 8-بت.
سجل الفحص Test Register هو عبارة عن دارة ماسك، تقوم بحفظ القيم الناتجة عن عمليات المقارنة التي تحصل ضمن وحدة الحساب والمنطق. تستطيع وحدة الحساب والمنطق أن تقارن بين عددين وتحدد إذا كانا متساويين أم لا، أو إذا كان أحدهما أكبر من الآخر. يستطيع أيضاً سجل الفحص أن يقوم بالاحتفاظ بالقيم الثنائية التي تنتج عن مرحلة الجمع الأخيرة في دارة الجامع، حيث يقوم السجل بحفظ هذه القيم في دارات

قلابات، ومن ثم يقوم مُرّمز التعليمات Instructions Decoder باستخدام هذه القيم من أجل اتخاذ القرارات.
يوجد 6 صناديق مكتوب عليها "ثلاث حالات "3-States"، وهي عبارة عن عوازل ثلاثية الحالة، وهذا يعني أنك تحصل على خرج هذه الدارات إما على قيمة (1) أو (0) أو لا تحصل على أي شيء. هذه الدارات تسمح بأن يتصل عدة مخارج بسلكٍ واحد، ولكن أحدها فقط سيستطيع أن ينقل قيمة (1) أو (0) إلى السلك.
سجل التعليمات Instruction Register ومرمز التعليمات يتولان مسؤولية التحكم بكل العناصر الأخرى.




البنية الأساسية لمكونات المعالج الصغري

هنالك جزء ليس واضح ضمن المخطط، وهو خطوط التحكم Control Lines الخارجة من مُرمز التعليمات، والتي تقوم بالمهمات التالية:
–إخبار السجل A بـ "مسك" Latch" القيمة الموجودة حالياً ضمن ناقل البيانات
–إخبار السجل B بـ "مسك" Latch" القيمة الموجودة حالياً ضمن ناقل البيانات

–إخبار السجل C بـ "مسك "Latch القيمة الموجودة حالياً على خرج وحدة الحساب والمنطق
–إخبار سجل عداد البرامج بـ "مسك" القيمة الموجودة حالياً على ناقل البيانات
–إخبار سجل العناوين بـ "مسك" القيمة الموجودة حالياً على ناقل البيانات
–إخبار سجل التعليمات بـ "مسك" القيمة الموجودة حالياً على ناقل البيانات
–إخبار عداد البرامج بأن يزيد قيمته
–إخبار عداد البرامج بأن يصفر قيمته
–تفعيل أحد العوازل ثلاثية الحالة الثلاث
–إخبار وحدة الحساب والمنطق ما هي العملية التي يجب أن يؤديها
–إخبار سجل الفحص بـ "مسك" بتات الفحص الخاصة بوحدة الحساب والمنطق
–تفعيل خط القراءةRD Line
–تفعيل خط الكتابةWR Line

يجب أن نذكر أمراً هاماً يتعلق بعمل المعالجات الصغرية، وهو دارات الدعم Support Circuitsفالمعالج الصغري بحاجة للعديد من الدارات المتكاملة التي تقوم بتنفيذ العديد من المهام المختلفة لتساعده وتدعم عمله الأساسي. فمثلاً، يحتاج المعالج الصغري لدارات إدخال وإخراجI/O Ports ، سواء كانت تفرعية أو تسلسلية، بالإضافة إلى الذواكر، والسجلات والعوازل التي ذكرناها سابقاً. ونظراً للأهمية، فإننا سنستعرض بشيءٍ من التفصيل الذواكر الأساسية المرتبطة بعمل المعالج الصغري، وبدون التوسع في شرح بقية دارات الدعم.

ذاكرة المعالج الصغريMicroprocessor Memory

قمنا في القسم السابق بتوضيح ناقل البيانات وناقل العناوين، وأيضاً خطوط القراءة والكتابة. هذه النواقل والخطوط تتصل بنوعين من الذواكر (بشكلٍ أساسي): ذاكرة الوصول العشوائي RAM وذاكرة القراءة فقط .ROM ولو عدنا للمثال البسيط الذي افترضناه، فإننا وضعنا قيمةً لعرض نواقل البيانات والعناوين هي 8-بت. فما هي فائدة معرفة هذه القيمة؟ في الواقع هذه القيمة أساسية جداً بعمليات تصميم المعالجات الصغرية، فهي تحدد عدد مواقع الذاكرة التي يستطيع المعالج التعامل معها وعنونتها. وبشكلٍ أساسي، فإن عدد مواقع الذاكرة التي يستطيع المعالج عنونتها هي:

n2

و (n) في العلاقة السابقة هي سعة نواقل البيانات للمعالج الصغري، وهذا يعني أن معالجاً بسعة نواقل بيانات قدرها 8-بت، فهو قادر على التعامل وعنونة (28) موقع في الذاكرة، أي 256 موقع في الذاكرة. وبالنسبة لمعالج يمتلك سعة نواقل قدرها 16-بت فإنه سيكون قادراً على عنونة 65536 موقع، ومعالج بسعة 32-بت سيكون قادراً على عنونة 4294967296 موقعاً في الذاكرة، ومعالج بسعة 64-بت سيكون قادراً على عنونة 18446744073709551616 موقع في الذاكرة.

بالنسبة لذاكرة القراءة فقطROM: Read-Only-Memory ، فهي ذاكرة لا يمكن للمستخدم الولوج إليها، حيث يتم برمجتها مع مجموعةٍ ثابتة من التعليمات، ويقوم ناقل العناوين بإخبار ذاكرة ROM ما هو البايت المناسب الذي يجب أن يتم وضعه على ناقل البيانات، وعندما يقوم خط القراءة بتبديل حالته، تقوم ذاكرة ROM بتمثيل البايت الذي تم اختياره على ناقل البيانات.

ذاكرة الوصول العشوائي RAM: Random-Access-Memory تتضمن بايتات المعلومات، حيث يستطيع المعالج الصغري الكتابة أو القراءة من هذه البايتات، وذلك بحسب حالة خطوط القراءة والكتابة، وأي منها تم تفعيله. أحد المشاكل الأساسية المتعلقة بذاكرة RAM هي أنها لا تستطيع الاحتفاظ بالمعلومات بعد انقطاع التيار الكهربائي، وهذا هو السبب الأساسي بحاجة الحواسيب لذواكر.ROM

للتوضيح أكثر: في الحواسيب المكتبية PC Desktop أو الحواسيب المحمولة التي نستخدمها في أيامنا هذه، فإن الـ ROM هي ما يعرف بالـ BIOS، أو نظام الإدخال والإخراج الأساسي، فعندما يبدأ المعالج الصغري بالعمل، فإنه يبدأ بتنفيذ العمليات الأساسية اعتماداً على التعليمات المحفوظة في الـ BIOS، حيث تعمل تعليمات الـ BIOS على أداء مهماتٍ أساسية مثل فحص جهوزية كافة العتاد الصلب Hardware الخاص بالآلة، ومن ثم تقوم بالذهاب إلى الأقراص الصلبة من أجل جلب قطاع التمهيد Boot Sector، وهو برنامج آخر هام لبدء عمليات التشغيل، وما يقوم به الـ BIOS هو قرائته ومن ثم وضعه على ذاكرة الـ RAM. بعد ذلك، يقوم المعالج الصغري بتنفيذ التعليمات الموجودة ضمن قطاع التمهيد اعتماداً على قرائتها من ذاكرة الـ RAM. بعد ذلك، يعمل قطاع التمهيد على إخبار المعالج الصغري بجلب شيء آخر من الأقراص الصلبة ووضعه على ذاكرة الـ RAM كي يتم تنفيذه.

إذاً، وبشكلٍ مختصر، فإن ذاكرة ROM تتضمن التعليمات الأساسية التي يحتاجها المعالج الصغري كي يعمل، بينما ذاكرة RAM تتضمن التعلميات قيد التنفيذ، حيث يتم قراءة التعليمات والمعلومات منها، بينما تقوم أقراص التخرين الصلبة بحفظ البرامج والمعلومات التي يتم تنفيذها وقرائتها لاحقاً من ذاكرة الـRAM.

التوضيح السابق هو اختصار لكيفية تنفيذ المعالج الصغري لكافة تعليمات نظام التشغيل.

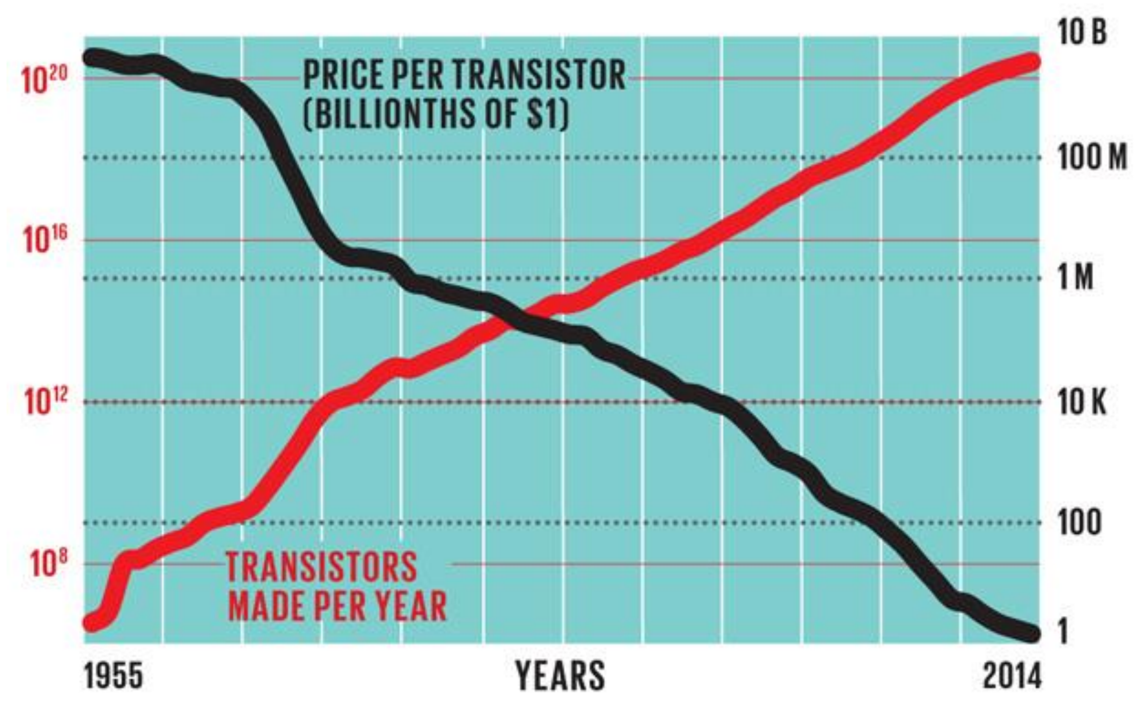
بالنسبة لموضوع برمجة المعالجات الصغرية من قبل المستخدمين (هذا الموضوع يهم مستخدمي المتحكمات الصغرية بشكلٍ أكبر)، فإن اللغة الأساسية للتعامل مع المعالجات الصغرية هي لغة التجميع Assembly Language والتي تعتبر أحد لغات البرمجة منخفضة المستوى كما ذكرنا في الجزء الثالث من السلسلة . طبعاً، فإن المعالج الصغري وبسبب بنيته المنطقية، فإنه لن يقبل التعليمات إلا على أعداد ثنائية مكونة من الأصفار والواحدات، ولكن عملية كافة الأرقام والأعداد والرموز على شكل بتات ثنائية سيكون أمراً شاقاً وصعباً، ولذلك فإنه يتم استخدام لغات البرمجة مثل لغة التجميع، وحديثاً يتم استخدام لغة C على نطاقٍ واسع من أجل برمجة المعالجات والمتحكمات الصغرية. طبعاً استخدام هذه اللغات هي لتسهيل المهمة على المبرمج، أما المعالج نفسه فلا يمكن له أن يفهم التعليمات المكتوبة بهذه اللغات، لذلك يتم تحويل البرامج المكتوبة باستخدام اللغات عالية المستوى إلى التعليمات الثنائية الموافقة باستخدام المفسرات والمترجمات، التي تتولى توليد التعليمات الثنائية اللازمة لعمل المعالج وفقاً للبرنامج المكتوب من قبل المبرمج.

الخلاصة :

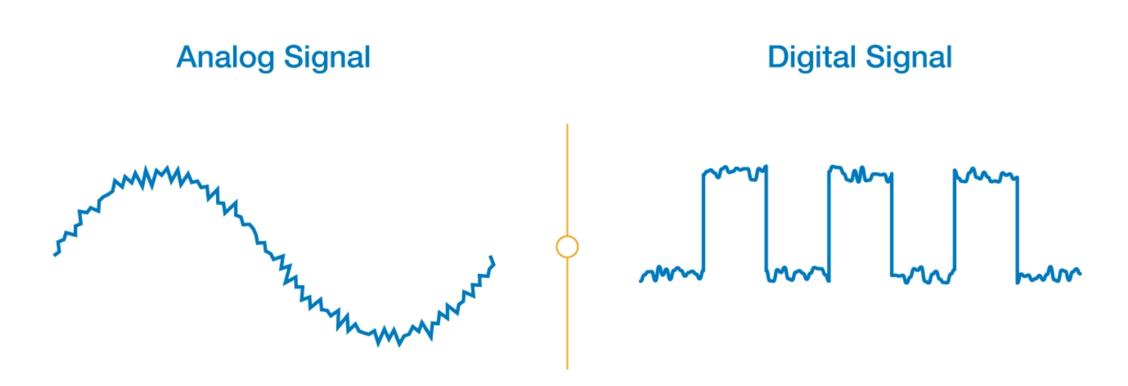
المعالجات الصغرية هي الوحدات الأساسية التي يتكون منها الحاسوب الرقمي الحديث، أياً كان شكله، سواء كان حاسوب مكتبي تقليدي، أم حاسوب محمول، أم جهاز لوحي، أو حتى هاتف ذكي. وتقوم المعالجات الصغرية بالتحكم بكافة الفعاليات الحاسوبية، وهي دماغه الأساسي. تتضمن المعالجات الصغرية مجموعة من الوحدات التي تقوم بتنفيذ مهام مختلفة (السجلات، الذواكر، وحدة الحساب والمنطق) كما تتضمن خطوط للقراءة والكتابة ونواقل للبيانات ونواقل للعناوين على مواقع الذاكرة. أخيراً، فإن المعالجات الصغرية تتميز بسمةٍ أساسية وهي سعة المعالج، أو دقته، أو عرض نواقل البيانات

الخاص به، وبناءً على ذلك، فإننا نميز بين معالجات 8-بت، أو 16-بت، أو 32-بت، أو 64-بت، وحالياً، فإن معظم الأجهزة الحاسوبية على مختلف أنواعها تستخدم معالجات 64-بت.

مُعدل إنتاج الترانزيستورات قد وصل إلى مستوياتٍ فلكية! انظر الشكل التالي :



## الاشارات التماثلية (التناظرية) والرقمية




العمل مع الإلكترونيات يشتمل على العمل مع كلا من الإشارات الرقمية والتناظرية، سواء كانت على شكل مدخلات أو مخرجات. مشاريعنا الإلكترونية التي نقوم بإنشائها تحتاج للتفاعل مع العالم الحقيقي التناظري بطريقة ما، لكن معظم المتحكمات الدقيقة (microprocessors)والحواسيب والوحدات المنطقية هي عبارة عن مكونات رقمية تماما. هذان النوعان من الإشارات يمكن تشبيههما بلغتين إلكترونيتين؛ بعض الإلكترونيات تتعامل بكلا اللغتين، بينما البعض الآخر يمكنه التعامل بلغة واحدة فقط منهما.

في هذا الدرس سنسلط الضوء على مبادئ الإشارات الرقمية والتناظرية، بما في ذلك أمثلة على كل منهما. سنتحدث أيضاً عن الدوائر والمكونات الرقمية والتناظرية.

- الإشارات التناظرية (Analog Signals)

الإشارات الإلكترونية تحديداً. الإشارات التي نتحدث عنها هي عبارة عن كميات تتغير بتغير الزمن (time-varying quantities) تحمل وتنقل أحد أنواع البيانات. في الهندسة الكهربية الكمية التي تتغير بتغير الزمن هي عادة الجهد (voltage) إن لم تكن كذلك فغالباً التيار .((current لذلك عندما نتحدث عن الإشارات فكر في الأمر كأنه جهد يتغير بتغير الزمن.

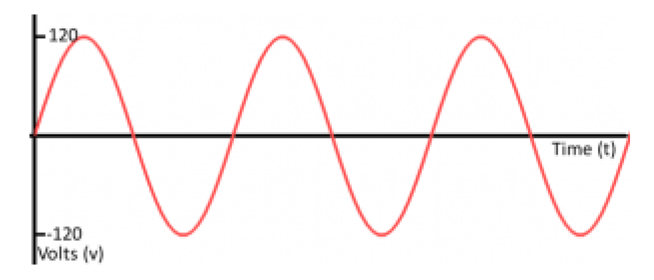
تنتقل الإشارات بين الأجهزة المختلفة من أجل إرسال واستقبال المعلومات، والتي ربما تكون فيديو، صوت أوي نوع من المعلومات المشفرة. عادة تنتقل الإشارات عبر الأسلاك، ولكن يمكن أيضاً أن تنتقل في الهواء عبر موجات الترددات الراديوية (radio frequency waves).

إشارات الصوت –على سبيل المثال- يمكن أن تنتقل بين بطاقة الصوت الخاص بحاسوبك وبين مكبر الصوت، بينما إشارات الإنترنت يمكن أن تنتقل عبر الهواء بين الكمبيوتر اللوحي والراوتر اللاسلكي

- الرسوم البيانية للإشارات التناظرية

لأن الإشارة تتغير بتغير الزمن، فمن المفيد أن نقوم بتمثيلها بيانياً حيث يتم تمثيل الزمن على المحور السيني(x-axis) ، أما الجهد فيتم تمثيله على المحور الصادي.(y-axis)

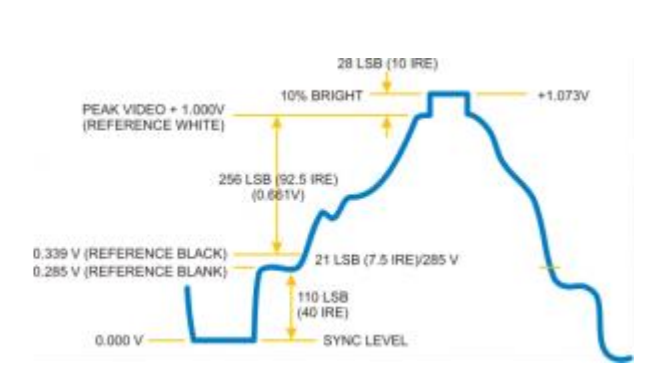
يعد النظر إلى الرسم البياني للإشارة غالباً أسهل الطرق لمعرفة نوعها تناظرية أم رقمية؛ الرسم البياني للجهد مقابل الزمن للإشارة التناظرية يجب أن يكون ناعماً ومستمراً.




بينما تقع هذه الإشارات في مدى محدود بين قيمتين قصوى ودنيا، إلا أنه ما يزال هناك عدد غير محدود من القيم المحتملة في هذا المدى. على سبيل المثال الجهد التناظري الآتي عبر مأخذ تيار الحائط يكون محصوراً بين قيمتين +V120 و -V120، لكن إذا نظرنا بدقة أكبر سنجد أن هناك عدد غير محدود من القيم يمكن أن تحملها الإشارة (مثل V64.4، V64.42، V64.424، وعدد لا محدود من القيم كلما نظرنا بدقة أكبر).

- مثال على الإشارات التناظرية

الفيديو أو الصوت غالباً ما يتم إرساله أو تسجيله باستخدام الإشارات التناظرية. الفيديو المركب (composite video) الآتي من مقبس RCA القديم على سبيل المثال هو عبارة عن إشارة تناظرية تقع في



مدى يتراوح غالباً بين V0 و V1.073. أي تغير صغير للغاية في الإشارة يكون له تأثير ضخم على لون أو موقع الفيديو.

إشارات الصوت النقية أيضاً هي إشارات تناظرية. الإشارات الآتية من مكبر الصوت تكون مليئة بالترددات والتوافقيات (harmonics) التناظرية، التي تجتمع معاً لتنتج أصوات رائعة.

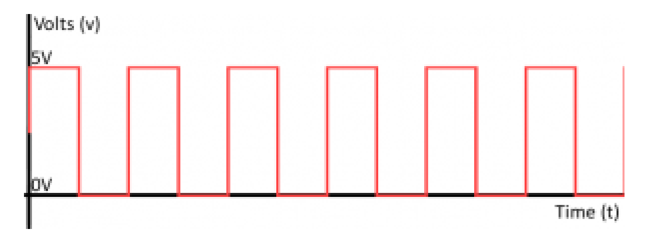
خصائص الاشارات التناظريه:

قيمها متغيره باستمرار.

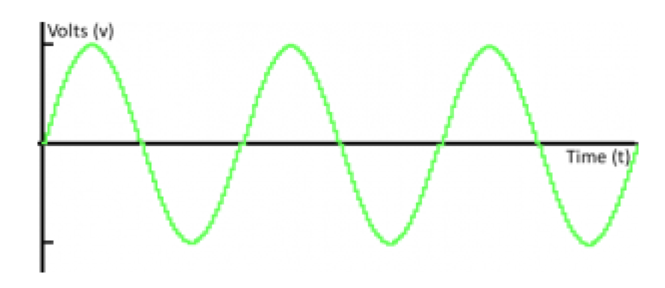
تتعرض للتشويش بسهوله .

- الإشارات الرقمية (Digital Signals)

الإشارات الرقمية لا بد أن يكون لها مجموعة محدودة من القيم المحتملة. عدد القيم المحتملة يمكن أن يكون أي قيمة بين اثنين أو أي عدد كبير آخر لا يساوي ما لا نهاية. معظم الإشارات الرقمية الشائعة لتكون أحد قيمتين اثنين- مثلاً إما V0 أو V5. الرسوم البيانية للجهد مقابل الوقت الخاصة بالإشارات الرقمية تكون على شكل موجات مربعة (square waves).




أي إشارة رقمية يمكن اعتبارها تمثيل متقطع لموجة تناظرية. عند النظر إلى الموجة في الرسم التالي عن بعد نجد أنها تبدو مثل إشارة تناظرية ناعمة، لكن عند النظر عن قرب نجد أن هناك خطوات متقطعة في الشكل الموجي للإشارة الرقمية.

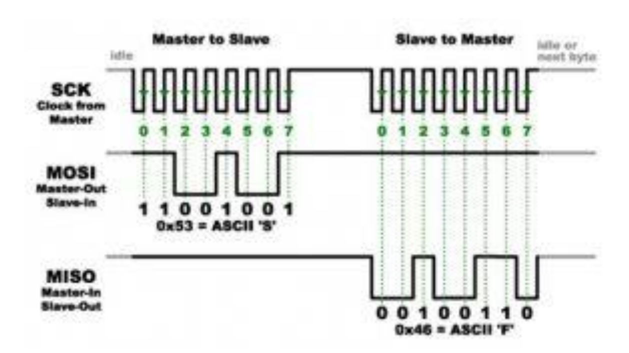



هذا هو أهم فرق بين الإشارات التناظرية والرقمية؛ الإشارات التناظرية ناعمة ومستمرة، بينما الإشارات الرقمية متقطعة ومتدرجة ومربعة.

- مثال على الإشارات الرقمية

ليست جميع إشارات الفيديو والصوت إشارات تناظرية. بعض الإشارات القياسية مثل واجهة الوسائط عالية الوضوح (HDMI) لنقل الفيديو (والصوت أيضاً)، وكذلك واجهاتMIDI ، I2S، AC’97لنقل الصوت جميعها تعتمد على الإشارات الرقمية.

جميع عمليات التواصل بداخل الدوائر المتكاملة (integrated circuits) تكون عمليات رقمية. والواجهات مثل الواجهة التسلسلية (serial)، آي سكويرد سي (I2C)، والواجهة التسلسلية الطرفية (SPI) جميعها تنقل البيانات عن طريق تسلسل مشفر من الموجات المربعة.




الواجهة التسلسلية الطرفية (SPI) تستخدم العديد من الإشارات الرقمية لنقل البيانات بين الأجهزة.

والانترنت عباره عن اشارات رقميه

والهاتف المحمول يرسل ويستقبل اشارات رقميه من والى ابراج الشبكه

خصائص الاشارات الرقميه:

دقه فى القياس عاليه جداً..

قابليتها للتشويش قليله جداً

ونستنتج من ما سبق ان :

الـ analog : هو كل اله تعتمد عليها فى قراءه بيانات معينه على مؤشر يتحرك على scaleمعين مثلا يتم تقسيمه بطريقه تتناسب مع الشئ المراد قياسه من تياراو فولت او قدره مثل محرك السيرفو .

الـ digital : هى كل اله تعتمد فى قراءه بياناتك على 0 و 1 اي تشغيل او اطفاء

مقارنة سريعة ..

مميزات الاشارات الـ تناظرية :

تغير قيمه باستمرار يفيد فى اخذ القراءات بعض المستشعرات و المكونات المستخدمه للاجهزه المطلوبه منها تستقبل الموجات التناظريه بتكون تكلفتها قليله واستخدامه بسيط

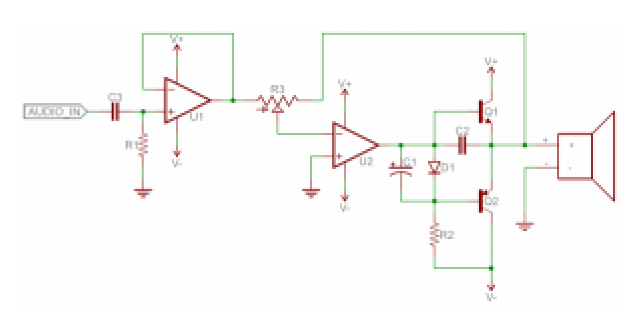
العيوب : تتاثر بالمسسافات حيث كلما بعدت المسافه تضعف الاشاره لذلك نستخدم المكبرات لتكبير الاشاره وتقويتها ,الـ noise فى الاشاره يشوش اجهزه الاستقبال بذلك يصدر صوت عن الاجهزه

مميزات الـ digital : قيمها ثابته لا تتغير

عيوب الـ digital : المكونات المستخدمه لها تكون شبه مكلفه واستخدامه يكاد يكون صعب

- الإلكترونيات التناظرية (Analog Electronics)

معظم المكونات الإلكترونية الأساسية مثل المقاومات (resistors)، المكثفات (capacitors)، المستحثات (inductors)، الديودات (diodes)، الترانزستورات (transistors)، والمضخمات العملياتية (operational amplifiers) جميعها تناظرية بطبيعتها. والدوائر التي يتم تكوينها من هذه المكونات فقط غالباً تكون دوائر تناظرية.



الدوائر التناظرية غالباً ما تحتوي على تجميعات معقدة من المضخمات العملياتية، المقاومات، المكثفات وبعض المكونات الإلكترونية الأساسية الأخرى. هذا المثال لمضخم صوتي تناظري من الفئةB

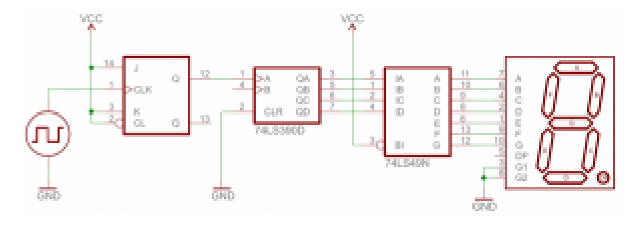
الدوائر التناظرية يمكن أن تكون معقدة للغاية مع الكثير من المكونات، ويمكن أيضاً أن تكون بسيطة للغاية مثل الجمع بين مقاومين في دائرة مجزئ الجهد (voltage divider).

بشكل عام الدوائر التناظرية تكون معقدة بشكل أكبر بكثير من الدوائر الرقمية التي يمكنها القيام بنفس المهمة. مثلاً لعمل مستقبل راديو تناظري أو شاحن بطارية تناظري نحتاج للكثير من العمل والجهد لتصميم دائرة تناظرية لهذا الغرض، بينما المكونات الرقمية متاحة لجعل هذه التصميمات أبسط وأسهل.

الدوائر التناظرية غالباً ما تكون أكثر عرضة للتشويش (noise) (اختلافات صغيرة غير مرغوب فيها في الجهد). الاختلافات البسيطة في مستوى الجهد الخاص بإشارة تناظرية يمكن أن ينتج عنها أخطاء أثناء عملية معالجتها.

- الإلكترونيات الرقمية (Digital Electronics)

الدوائر الرقمية تعمل اعتماداً على إشارات رقمية متقطعة. هذه الدوائر عادة ما تُصنع باستخدام تجميعات من الترانزستورات (transistors) والبوابات المنطقية (logic gates)، وعلى المستوى الأعلى المتحكمات الدقيقة (microcontrollers) أو رقاقات الحوسبة الأخرى (computing chips). معظم المعالجات (processors) سواء كانت كبيرة مثل المستخدمة في الحواسيب، أو مجرد متحكمات دقيقة تعمل بشكل رقمي.




في الدائر الرقمية يتم استخدام مكونات مثل البوابات المنطقية أو دوائر متكاملة رقمية أخرى أكثر تعقيداً وعادة يتم تمثيلها بمستطيلات يبرز منها أطراف مميزة بأرقام أو حروف.

تعتمد الدوائر الرقمية على نظام ثنائي (binary) لإرسال واستقبال الإشارات الرقمية. هذا النظام يخصص قيمتي جهد مختلفتين كمستويين منطقيين (logic levels) مختلفين جهد مرتفع عادة 5V، 3 V أو 1.8 Vليمثل إحدى القيم، وجهد منخفض (عادة يكون 0 V)لتمثيل القيمة الأخرى.

بالرغم من أن الدوائر الرقمية أسهل في تصميمها إلا أنها أغلى إلى حد ما من الدوائر التناظرية التي تقوم بنفس المهمة.

- الجمع بين التناظرية والرقمية

ليس من النادر أن نرى دائرة تجمع بين مكونات رقمية وتناظرية معاً. بالرغم من أن المتحكمات الدقيقة هي مكونات رقمية بشكل كامل إلا أنها عادة ما تحتوي على دوائر داخلية تمكنها من التفاعل مع الدوائر التناظرية (المحولات التناظرية الرقمية، تعديل عرض النبضة، والمحولات الرقمية التناظرية).

المحول التناظري الرقمي (analog-to-digital converter (ADC)) يسمح للمتحكم الدقيق بالاتصال بالمستشعرات التناظرية (مثل الخلايا الضوئية (photocells) أو مستشعرات الحرارة) عن طريق قراءة جهدها التناظري. المحول الرقمي التناظري (digital-to-analog converter) الأقل شيوعاً يسمح للمتحكم الدقيق بإنتاج جهد تناظري، وهذا يفيد عند الحاجة لإصدار صوت.

# تعديل عرض النبضة

**PWM** اختصار ل **P**ulse **W**idth **M**odulation هي تقنية تسمح بالتحكّم بقيمة تماثلية رقمياً (controlling an analog value digitally).اي التحكم يكون رقمي لصناعه موجه مربعه او مستطيلة (Square wave)، والاشارة تتذبذب بين التفعيل وإلغاؤه (LOW-HIGH)

الفكرة تكمن في التحّكم بتردد النبضات pulses في الدورة الواحدة، بمعنى انه يتم التبديل بين التردد العالي والتردد المنخفض بسرعة معيّنة بحيث ان الناتج النهائي يكون قيّمة بينهم. الإشارة الرقمية لها قيمتين كما هو معروف 0 و 1 ففي حالة لو كان الجهد 12 فولت مثلاً، 0 ≡ 0 فولت و 1 ≡ 12 فولت، بينما الإشارة التماثلية هي قيمة بين الصفر والـ 12 فولت.

الـ duty cycle أو ما يعرف بدورة الخدمة للإشارة الرقمية هي نسبة الجهد العالي في دورة واحدة من الإشارة، وهي قابلة للبرمجة أو التعديل ومعرّفة بالعلاقة الرياضية:

$$d = t_high / (t_high + t_low)$$

وهي عبارة عن نسبة مئوية يمكن استخدامها للربط بين الجهود

$$v_out = d \times v_in$$

وبهذا الشكل هذا يمكن التحكم في قيمة الجهد الثابت .باستخدام هذه الفكرة البسيطة نستطيع التحكم بالـ dc motors أو بشدّة الإضاءة الخاصة بـ led معيّن باستخدام مصدر جهد ثابت dc، وغيرها من التطبيقات. الجهد الناتج هو قيمة ما بين أعلى وأقل قيمة للجهد الخاص بالمصدر على حسب نسبة دورة الخدمة.

لو لدينا مصدر جهد خمسة فولت وقمنا بعمل موجة منه عن طريق المايكروكنترولر مثلا هذه الموجة قد تجعل معدل جهد الخرج أقل من خمسة فولت ( 4 فولت أو 3 أو 2.4 أو .. كما نريد! ) ومن المعلومات المهمة الخاصة بالموتور dc أنه كلما قل الفولت الداخل إليه كلما قلت سرعته

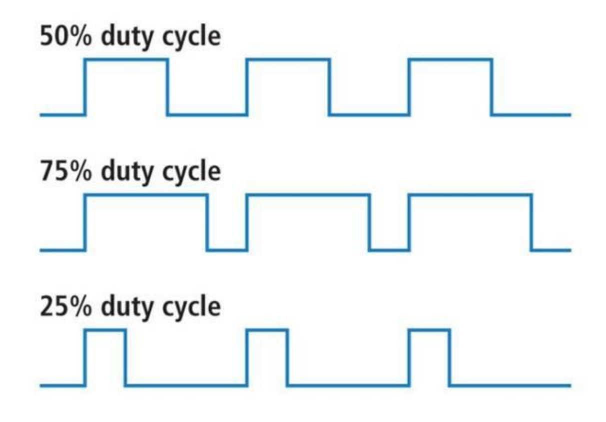
ملخص ما تحدثنا به , الطرق الشائعة لاستخدام تعديل عرض النبضة التحكم في مستوى إضاءة ديودات RGB المضيئة أو التحكم في اتجاه مُحرك السيرفو (servo motor). من الممكن تحقيق مدى واسع من النتائج في كلا التطبيقين، لأن تعديل عرض النبضة يسمح لنا بتغيير كمية الوقت الذي تكون في الإشارة ذات قيمة مرتفعة (high). يُمكن أن تكون الإشارة خلال أي وقت على إحدى حالتين: إما مرتفعة = high غالباً= أو منخفضة low (أرضي (ground))، لكن باستخدام تعديل عرض النبضة يمكننا تغير النسبة المئوية للزمن الذي تكون فيه الإشارة مرتفعة خلال فترة زمنية مُحددة.

# دورة العمل (Duty Cycle)

هو المسؤول عن الجهد الناتج عن عملية PWM وهو عبارة عن نسبة مئوية . فمثلا لو كان لدينا جهد المصدر خمسة فولت وجعلنا الـ duty cycle بخمسين في المئة 50% فإن معدل الفولت الناتج سيكون 2.5 فولت . ولو جعلنا الـ duty cycle بخمسة وعشرين في المئة 25% فإن معدل الفولت الناتج سيكون 1.25 فولت وهكذا ...

يُطلق على الوقت الذي تكون قيمة الإشارة فيه مرتفعة اسم "وقت التشغيل (on time) للتعبير عن كمية "وقت التشغيل" نستخدم مُصطلح دورة العمل. ويُعبر عن دورة العمل بنسبة مئوية. تصف دورة العمل النسبة المئوية للزمن الذي تكون فيه قيمة الإشارة مرتفعة خلال دورة زمنية مُحددة. هذه الدورة الزمنية (أو الزمن الدوري) تساوي مقلوب تردد الشكل الموجي.

فإذا كانت قيمة الإشارة مرتفعة لنصف الوقت ومنخفضة للنصف الآخر، نقول أن دورة العمل لهذه الإشارة الرقمية تساوي 50%، ويمكن تمثيلها بموجة مربعة ( square wave) مثالية. وإذا كانت قيمة دورة العمل أكبر من 50% فهذا يعني أن الإشارة الرقمية تكون ذات قيمة مُرتفعة لمدة أطول من كونها ذات قيمة منخفضة، والعكس صحيح في حالة كانت قيمة دورة العمل أقل من 50%. الرسوم البيانية التالية توضح هذه الحالات الثلاث أمثلة على دورة عمل بقيم 50%، 75%، 25% من أعلى إلى أسفل:



عندما تبلغ قيمة دورة العمل 100% فهذا يعني أن قيمة الجهد ثابتة عند V5 (قيمة مرتفعة)، بينما عندما تكون قيمة دورة العمل 0% فهذا يعني أن الإشارة مُؤرضة.

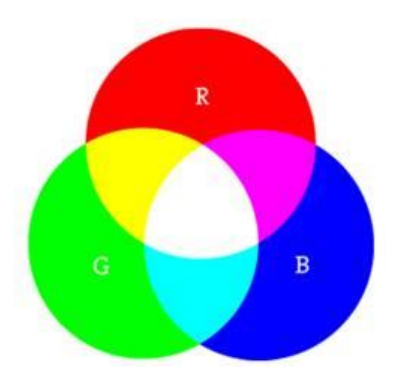
أمثلة:

يُمكنك التحكم في شدة إضاءة الديودات المضيئة عن طريق تعديل قيمة دورة العمل.



استخدام تعديل عرض النبضة للتحكم في شدة إضاءة الديودات المضيئة إذا كان لدينا ديود مضيء RGB (يصدر الألوان: أحمر (red)، أخضر (green)، أزرق (blue)) يُمكننا التحكم في كمية كل لون من الألوان الثلاثة ضمن مزيج الألوان عن طريق إعتام كل منها بدرجات متفاوتة.

القواعد الأساسية لمزج الألوان الرئيسية




إذا كانت درجات جميع الألوان متساوية يكون الناتج ضوء أبيض ذا سطوع متفاوت. أما مزج الأزرق مع الأخضر بالتساوي يُعطي لون أزرق مائل للخضرة (teal). ولنأخذ مثالاً أكثر تعقيداً: جرب أن تجعل دورة العمل للأحمر 100% وللأخضر 50% وللأزرق 0%، وسيكون اللون الناتج هو اللون البرتقالي.



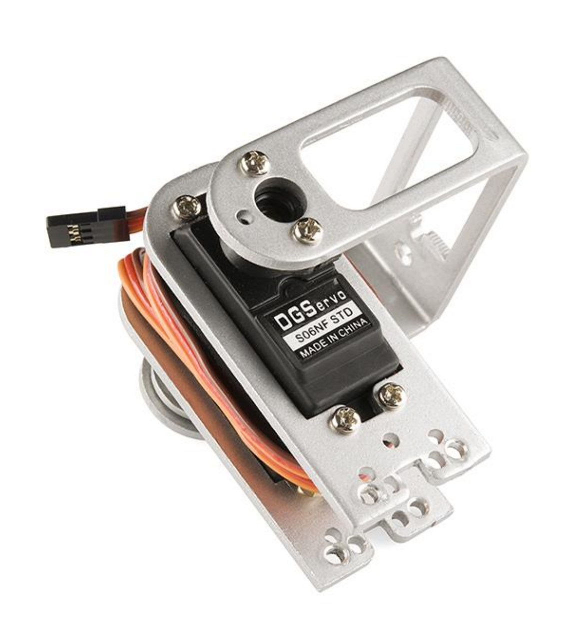
يُمكن استخدام تعديل عرض النبضة للتحكم في لون ديودات RGB

يجب أن يكون تردد الموجة المربعة كبيراً بدرجة كافية عند التحكم في الديودات المضيئة للحصول على شدة السطوع المناسبة. في حال كانت دورة العمل تساوي 20% والتردد 1Hz سيبدو لعينيك أن الديود الضوئي يضيء ويطفئ، بينما إذا كانت دورة العمل تساوي 20% والتردد 100Hz فسيبدو الديود الضوئي مضيء باستمرار لكن أعتم مما هو عليه عندما تكون دورة العمل 100%. كذلك من الضروري ألا يكون الزمني كبيراً إذا كنت تسعى إلى التحكم في شدة سطوع (درجة إعتام) الديودات الضوئية.

من الممكن كذلك استخدام تعديل عرض النبضة للتحكم في زاوية مُحرك سيرفو متصل بجسم ميكانيكي مثل ذراع الروبوت. حيث تحتوي مُحركات السيرفو على محور يُمكن جعله يدور إلى وضعية مُعينة باستخدام خط التحكم (control line) الخاص به. بعض مُحركات السيرفو يُمكن أن تدور في مدى يصل إلى 180 درجة.

تكون قيم التردد/الزمن الدوري مُحددة للتحكم في مُحرك سيرفو مُعين. بالنسبة لمُحرك السيرفو النموذجي يُتوقع أن يتم تحديثه كل ms20 باستخدام نبضة تتراوح بين ms1 و ms2، أو -بطريقة أخرى- بقيمة دورة عمل تتراوح بين 5% إلى 10%مع موجة ذات تردد Hz50. باستخدام نبضة ms1.5 يكون مُحرك السيرفو في موضعه الطبيعي عند 90 درجة، وباستخدام نبضة ms1 تكون وضعية مُحرك السيرفو عند زاوية 0، أما باستخدام نبضة ms2 يكون مُحرك السيرفو عند 180 درجة. ويُمكنك الحصول على مدى كامل من الحركة عن طريق تحديث المُحرك بقيم بين القيم السابقة.

الصورة التالية تمثل استخدام تعديل عرض النبضة لجعل مُحرك السيرفو في وضعية 90 درجة.



# انظمة العد



انظمة العد Numeral system هي طريقة عرض الأعداد برموز محددة والتعامل معها للتعبير عن قيمتها وكيفية تطبيق العمليات الحسابية عليها.وتستخدم أنظمة عد مختلفة لعرض الأعداد. فمثلاً العددين $(2A)_{16}$ و $(52)_8$ يعنيان نفس القيمة $(42)_{10}$ ولكن بطريقة عرض مختلفة. طريقة عرض الأعداد بأنظمة العد مختلفة هو نفس طريقة عرض الكلمات في اللغات المختلفة فمثلاً الكلمة cheval كلمة فرنسي والكلمة equusكلمة لاتينية والكلمة horse كلمة إنجليزية لهم نفس المعنى "حصان."

مثلما نستخدم الرسوم (الحروف) لإنشاء الكلمات في اللغة، كذلك نستخدم (الأرقام) لعرض الأعداد. وكما نعلم فإن عدد الحروف في أي لغة محدد لذا نعيد تكرارها لإنشاء

كلمات جديدة ومتعددة. كذلك الامر مع الأرقام فعددها محدود (على سبيل المثال، في النظام العشري هناك 10 أرقام فقط ، وفي النظام الثنائي عددين فقط) مما يحتم علينا تكرارها لإنشاء الأعداد. تعرف أنظمة العد.

في عالم الحواسيب والأنظمة الرقمية، يتم التعامل مع المعطيات والبيانات وفقاً لتمثيلٍ وطريقةٍ خاصة. هنا علينا أن نتذكر أمراً هام، وهو أن الدارات الحاسوبية المسؤولة عن معالجة المعطيات، وتخزينها، ونقلها، وحتى إرسالها لوحدات الإظهار(الشاشات مثلا)، لا تستطيع أن تفهم إلا نمطاً واحداً من المعطيات: المعطيات الثنائية، أي المعطيات التي تكون على شكل سلاسل من الأصفار والواحدات. بمعنى آخر، فالحاسوب يمتلك لغته الخاصة، التي تتخاطب أجزائه عبرها، وكي يستطيع فهم الأوامر التي نرسلها إليه باللغة التي نفهمها نحن، يجب أن يتم تحويل المعلومات والمعطيات إلى لغته الخاصة به، ودون ذلك، لن يستطيع الحاسوب تنفيذ عمليات المعالجة وأي نمط آخر من الأوامر قد نطلبها منه.

بهذا السياق، فإن التعامل مع الحواسيب والأنظمة الرقمية يتطلب معرفة بأنظمة أعداد معينة، تختلف قليلاً عن أنظمة الأعداد التي نتعامل معها بالحياة اليومية، ولكنها ترتبط بها، حيث يمكن التحويل من هذه الأنظمة إلى النظام الذي نفهمه بحياتنا اليومية، والعكس أيضاً صحيح.

من ناحيةٍ أخرى، وبما أن العمليات التي تجري ضمن الدارات الحاسوبية والأنظمة الرقمية تعتمد على الجبر البولياني، فإن الأعداد التي سيتم التعامل معها يجب أن تخضع أيضاً لشروط الجبر البولياني، وأي نمط معلومات يتم إدخاله للحاسوب، يجب أن يتم تحويله للصيغ التي يفهمها الجبر البولياني، كي تتمكن الدارات الحاسوبية من التعامل معها، ومعالجتها، ومن ثم تقوم بإعادة تحويلها للنمط الذي نفهمه نحن كبشر.

إذاً، فإننا سنستعرض ضمن في هذا الدرس أنظمة العد التالية، والتي تعتبر حجر أساس لأي شخص يريد الدخول لعالم الأنظمة الرقمية والدوائر الالكترونية :

1 -نظام الأعداد العشري Decimal System

2 -نظام الأعداد الثنائي Binary System

3 -نظام الأعداد الثماني Octal System

4- نظام الأعداد الستة عشري Hexa-Decimal System

يجدر بنا التنويه هنا إلى أمرٍ هام وهو أساسيات أي نظام أعداد، فكل نظام أعداد يتصف بشكلٍ أساسي بما يلي:

- أساس النظام: وسنعطيه الرمز(n)
- مكونات النظام: وهي الأعداد من (0) وحتى(n-1)

بناءً على الأساسيات البسيطة السابقة، أصبح الآن بإمكاننا البدء بتوضيح أنظمة العد ودورها في العالم الرقمي، ومن نظام الأعداد العشري سنبدأ .

اولا : نظام الأعداد العشريDecimal System

نظام الأعداد العشري هو أبسط أنظمة العد من حيث سهولة الفهم لنا كبشر. فهو نظام الأعداد الذي نستخدمه في الحياة اليومية، والذي نستطيع تطبيق عليه العمليات الجبرية بالشكل التقليدي: أي الجمع والطرح والضرب والقسمة.

أساس النظام العشري هو العدد (10)، وبالتالي فإن مكوناته ستكون من (0) وحتى (10-1) أي من (0) وحتى (9). بالنسبة لنظام الأعداد العشري، فإن أي رقم فيه يمكن تمثيله باستخدام كثيرات الحدود. ومن دون الخوض بالتفاصيل، سنأخذ المثال التالي كي نفهم أكثر كيفية بناء العدد العشري.

ليكن لدينا العدد (843)، فإنه فعلياً عبارة عن مجموع جداءات كل عدد، مضروباً بالمرتبة العشرية الموافقة. فالعدد (8) سيكون مضروباً بالمرتبة العشرية الثانية، أي (10^2) وهي 100، والعدد (4) سيكون مضروباً بالمرتبة العشرية الأولى، أي (10^1) وهي العدد 10، أما العدد (3) فسيكون مضروباً بالمرتبة العشرية الصفرية، أي (10^0) وهي تساوي الواحد، ومع جمع الحدود السابقة، نستطيع الحصول على القيمة الكلية للعدد، وهي 843. تدعى المرتبة العشرية التي يتم الضرب بها بالـ "وزن".

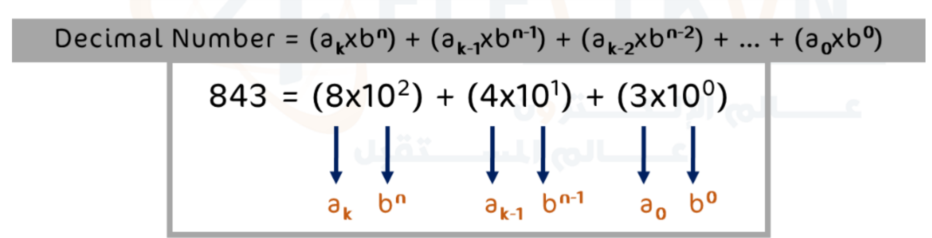
يوضح الشكل التالي نظام الأعداد العشري :

نظام الأعداد العشري

أساس النظام Base: 10

مكونات النظام Components: {0,1,2,3,4,5,6,7,8,9}

تمثيل الأعداد: كثيرات الحدود




ثانيا : نظام الأعداد الثنائي Binary Number System

نظام الأعداد الثنائي، فعلياً، هو أساس الثورة الرقمية بالكامل، وأساس عمليات المعالجة الحاسوبية كلها. بشكلٍ بسيط، فإن هذا النظام يتألف من عددين فقط : 0 و 1. وبالتالي فإن أساس النظام هو (2) ومكونات النظام هي {0,1}.

كي تصبح الصورة أوضح، فإن أي نمط من أنماط المعلومات، يتم ترميزه في نظام الأعداد الثنائي باستخدام سلسلة من الأصفار والواحدات. فكر بأي نمط من أنماط المعلومات: حرف، كلمة، صورة، فيديو، كلها يتم تمثيلها في نظام الأعداد الثنائي باستخدام سلاسل طويلة من الأصفار والواحدت، ولا شيء سواها.

تكمن أهمية النظام الثنائي، من كونه نظام الترميز المستخدم لتمثيل المعلومات والمعطيات ضمن الدارات الحاسوبية والرقمية، فأي معلومة، يتم تحويلها بشكلٍ أساسي إلى نظام الأعداد الثنائي، ومن ثم يتم تتم معالجتها ضمن الحاسوب، وأخيراً يتم تحويل ناتج المعالجة للشكل الذي نستوعبه، ويتم إظهاره. هذا – ببساطة – مبدأ عمل كافة الأجهزة الحاسوبية، بدءاً من الحواسيب المنزلية، إلى الحواسيب المحمولة، وصولاً للهواتف الذكية والحواسيب اللوحية بالطبع، هنالك قواعد للتحويل بين أنماط المعطيات والمعلومات لتصبح متناسبة مع النظام الثنائي، وكذلك طرق لتحويل المعلومات والمعطيات من النظام الثنائي إلى الأنماط الأخرى، لتصبح قابلةً للفهم والاستيعاب من قبلنا.

## ثالثا : النظام الثنائي البت والبايت Bit and Byte

في النظام الثنائي، فإن كل رقم فيه يعرف باسم "البت" bit ، وهو اختصار لكلمة "رقم ثنائي ."Binary Digit البت، هو أصغر واحدة لترميز المعلومات في الحاسوب، وكل الواحدات الأخرى التي نستعملها يومياً هي من مضاعفات البت.

البايتByte ، هو عبارة عن سلسلة من (8) بتات، وبالتالي، فإن أي سلسلة من 8 أرقام ثنائية تمثل بايتاً واحداً. أيضاً، يتم استخدام مضاعفات واحدة البايت في الحياة اليومية من أجل تمثيل المعطيات والمعلومات. لنأخذ مثال توضيحي، ولنتخيل صورةً ذات حجم 512 كيلوبايت. ما الذي نقصده بأن حجم الصورة هو 512 كيلوبايت؟

في الواقع، فإن ما نقصده بالضبط، هو أن الصورة نفسها، والتي تضم طيفاً واسعاً من الألوان، والخطوط، وغيرها، قد تم ترميزها باستخدام مساحة قدرها 512 كيلوبايت، وبما أن البايت نفسه هو (8) بت، والكيلوبايت هو عبارة عن 1024 بايت، فهذا يعني أن الصورة التي يبلغ حجمها 512 كيلوبايت، قد تم ترميزها باستخدام عدد من الأصفار والواحدات يبلغ: 512*1024*8 = 4194304 بت! أي 4194304 عدد ثنائي! الآن يمكنكم أن تتخيلوا كمية الأصفار والواحدات الهائلة التي تتضمنها الذواكر الحالية والتي تبلغ بشكلٍ وسطي 1 تيرابايت من المعطيات!

ولكن، ما الذي يجعل النظام الثنائي مميزاً؟

الإجابة على السؤال السابق تنحصر كما يلي: أولاً، يمكن عبر النظام تمثيل أي عدد من أي نظام عددي (وهو ما سنأتي على توضيحه). ثانياً، النظام الثنائي هو النظام العددي الذي يمكن تطبيق قواعد الجبر البولياني عليه بشكلٍ مباشر، وبالتالي تنفيذ كافة العمليات المنطقية اللازمة من أجل تكوين الدارات الحاسوبية والرقمية. هذا الأمر، هو ما جعل نظام الأعداد الثنائي أساس الثورة الرقمية، وأساس نظم ترميز المعلومات.

نظام الأعداد الثنائي

أساس النظام Base: 2

مكونات النظام Components: {0,1}

تمثيل الأعداد: سلاسل من الأصفار والواحدات

Binary Number = 01100101100011 0100.....

- البت الواحد = 0 أو 1
- البايت الواحد: 8 بت

رابعا : نظام الأعداد الثمانيOctal Number System

نظام الأعداد الثماني هو نظام الأعداد الذي أساسه العدد (8)، وهذا يعني أن مكونات هذا النظام ستكون: {0,1,2,3,4,5,6,7}. لا يستخدم نظام الأعداد الثماني بكثرةٍ في أيامنا هذه، واقتصرت استخداماته على البدايات الأولى لظهور الحواسيب.

خامسا : نظام الأعداد الستة عشريHexadecimal Number System

يمتلك نظام الأعداد الستة عشري أساساً هو العدد (16)، وبالتالي فإن مكونات هذا النظام هي الأعداد التالية: {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}. وبهدف التمييز بينه وبين النظام العشري، تم ترميز الأرقام من 10 وحتى 15 في

النظام الستة عشري بالأحرف التالية: A, B, C, D, E, F حيث تشير هذه الأحرف للأرقام 10,11,12,13,14,15 على التوالي.

- التحويل بين أنظمة العد

قبل الحديث عن استخدامات أنظمة العد، يجب أن نعلم أمراً هاماً حولها: يمكن التحويل بين أنظمة العد وبسهولة، وذلك وفقاً لقواعد معينة. الهدف من عملية التحويل هذه هو نقل المعطيات من الصيغ التي نتعامل معها ونفهمها مباشرةً (مثل النظام العشري) إلى الصيغ التي يتعامل معها الحاسوب ويفهمها (مثل النظام الستة عشري والنظام الثنائي). وبالعودة للجبر البولياني، فإننا سنجد مثلاً أن تنفيذ العمليات الرياضية التابعة للجبر البولياني لا يمكن أن يتم إلا على المعطيات المنطقية التي تحمل أحد حالتين: الحالة الصحيحة True أو الحالة الخاطئةFalse ، وبالنظر لأنظمة العد، فإننا نستطيع أنه لا يوجد نظام عد متوافق مع هذه العمليات إلا النظام الثنائي، فعبر إسناد قيمة "1" للحالة الصحيحة، وقيمة "0" للحالة الخاطئة، أصبح بإمكاننا تنفيذ العمليات الرياضية في الجبر البولياني وبالشكل الكامل.

ينبغي لنا أن نقول أننا لن نستعرض كافة المفاهيم المتعلقة بالتحويل بين أنظمة العد بشكلٍ مفصل، بل سنكتفي بالإشارة لأهم ما يتعلق بها، وإعطاء بعض الأمثلة التوضيحية عنها.

1 -عمليات التحويل الخاصة بالنظام العشري

بالنسبة للنظام العشري، فإن عملية تحويل أي عدد بالنظام العشري، إلى العدد الموافق له بنظام عد آخر، هي أسهل عمليات التحويل، حيث تتم بالتقسيم على أساس النظام، مع أخذ باقي القسمة بعين الاعتبار. فلو أردنا التحويل من النظام العشري للنظام الثنائي، نأخذ الرقم العشري المراد تحويله، ونقوم بتقسيمه على العدد "2"، وذلك حتى تصبح عملية القسمة غير ممكنة، ومن ثم نقوم بأخذ باقي القسمة من الأسفل للأعلى، ونحصل بالتالي على العدد العشري ممثلاً بالنظام الثنائي. هذا المفهوم ينطبق أيضاً على النظام الثماني والستة عشري وبنفس الطريقة تماماً.

# التحويل من النظام العشري للأنظمة الباقية

### من العشري للستة عشري

ليكن العدد العشري 56، ما هو مقابله الستة عشري؟

56 / 16 = 3 %=8
3 / 16 = 0 %=3

$(56)_{10} = (83)_{16}$

تتم عملية التحويل بالتقسيم على العدد 16حتى يصبح العدد غير قابل للقسمة، ويتم أخذ الناتج من باقي القسمة.

### من العشري للثماني

ليكن العدد العشري 56، ما هو مقابله الثماني؟

56 / 8 = 7 %=0
7 / 8 = 0 %=7

$(56)_{10} = (07)_{8}$

تتم عملية التحويل بالتقسيم على العدد 8 حتى يصبح العدد غير قابل للقسمة، ويتم أخذ الناتج من باقي القسمة.

### من العشري للثنائي

ليكن العدد العشري 56، ما هو مقابله الثنائي؟

56 / 2 = 28 %=0
28 / 2 = 14 %=0
14/2 = 7 %=0
7/2 = 3 %=1
3/2 = 1 %=1
1/2 = 0 %=1

$(56)_{10} = (111000)_{2}$

تتم عملية التحويل بالتقسيم على العدد 2 حتى يصبح العدد غير قابل للقسمة، ويتم أخذ الناتج من باقي القسمة.

هنالك ملاحظة هامة هنا، وهي أنه على الرغم من أن الأرقام المستخدمة في النظامين الثماني والستة عشري، هي مشابهة للأرقام في النظام العشري، إلا أنها ليست نفسها! فالرقم (56) الثماني ليس نفسه (56) العشري، والرقم (56) الستة عشري ليس نفسه (56) العشري أو حتى (56) الثماني. توضيح هذه النقطة سيكون عند الحديث عن علاقة النظام الثنائي بالأنظمة الأخرى.

الآن، علينا توضيح كيفية الانتقال من أحد الأنظمة للنظام العشري. وهنا علينا تذكر أمر هام جداً، وهو كيفية ترميز النظام العشري، وقد قلنا سابقاً أن النظام العشري هو نظام الأعداد الذي يتم التعبير عن أي عددٍ فيه باستخدام مبدأ كثيرات الحدود Polynomials. الآن، ومن أجل تحويل أي عدد للنظام العشري، فإن كل ما علينا القيام به هو استخدام قاعدة كثيرات الحدود، مع استبدال أساس القوة التي نضرب بها، بأساس النظام العددي نفسه، فإذا كان الرقم المراد تحويله بالنظام الثنائي، فإن أساس قوة العدد سيكون (2)، وإذا كان العدد بالنظام الثماني، فإن أساس القوة هو (8)، وإذا كان بالنظام الستة عشري، فإن أساس القوة هو العدد (16). وسنستخدم نفس المثال السابق، ولكن معكوساً، أي أننا نريد استخدام الأعداد السابقة من أجل معرفة مقابلها العشري.

# التحويل إلى النظام العشري

| من السنة عشر للعشري | من الثماني للعشري | من الثنائي للعشري |
| --- | --- | --- |
| $(83)_{16} = (8x16^0) + (3x16^2)$<br>$(83)_{16} = 8 + 48$<br>$(83)_{16} = (56)_{10}$ | $(07)_8 = (0x8^0) + (7x8^1)$<br>$(07)_8 = 0 + 56$<br>$(07)_8 = (56)_{10}$ | $(111000)_2 = (0x2^0) + (0x2^1) + (0x2^2) + (1x2^3) + (1x2^4) + (1x2^5)$<br>$(111000)_2 = 0 + 0 + 0 + 8 + 16 + 32$<br>$(111000)_2 = (56)_{10}$ |

للحصول على المقابل العشري، نستخدم قاعدة كثيرات الحدود، ونبدأ من أقصى خانة في اليمين، وصولاً لأقصى خانة في اليسار، مع الأخذ بعين الاعتبار قوة كل خانة

## 2 -التحويل من وإلى النظام الثنائي

بدأنا توضيحنا لعمليات التحويل بالنظام العشري، لأنه يمثل النظام العددي المألوف بالنسبة لنا، والذي يمكن أن تطبق عليه العمليات الجبرية كما نعرفها. الآن، سننتقل لتوضيح كيفية التحويل إلى نظام الأعداد الثنائي وبالعكس.

من أجل التحويل للنظام الثنائي، علينا تذكر أن النظام الثنائي يتكون من عددين فقط: الصفر "0" والواحد "1"، ويمكن التعبير عن أي رقمٍ من أي نظام عددي بالنظام الثنائي، عبر سلسلةٍ من الأعداد الثنائية. تدعى أقصى خانة من اليمين ضمن السلسلة الثنائية بالـ "البت الأقل أهمية" LSB بينما تدعى أقصى خانة من اليسار بالـ "البت الأكثر أهمية MSB". على سبيل المثال، فإن العدد العشري (45) يتم تمثيله بالنظام الثنائي بالسلسلة (101101). كيف يتم ذلك؟ علينا أن نتذكر أن العدد العشري يتم ترميزه باستخدام مبدأ كثيرات الحدود، وبناءً عليه، فإن كل خانة ضمن السلسلة الثنائية الممثلة للعدد العشري سيكون لها "وزن". وزن الخانة الثنائية يمثل القيمة الموضوعة فيها (صفر أو واحد) مضروبةً بالعدد 2 (أساس نظام العد) مرفوعاً للأس الموافق لترتيب الخانة. وفي حين أن الترتيب يبدأ من أقصى اليمين، فإن العدد الموضوع في الخانة ذات الترتيب "0" سيأخذ القوة (0)، والخانة التي تليها ستكون ذات معامل قوة قدره (1)، والتي تليها ستكون ذات معامل قدره (2)، وهكذا قدر ما نشاء. الآن، كيف نقوم بحساب السلسلة الثنائية الموافقة؟ حسناً، سنأخذ مثالاً وليكن العدد (45) العشري، ونريد أن نقوم بحساب الموافق العشري له.

ما علينا القيام به هو اتباع الخطوات التالية:

نقوم بكتابة سلسلة عددية، أساسها العدد (2)، وكل حد يختلف فيها عن الحد الذي يسبقه بقيمة الأس. تبدأ هذه السلسلة بالعدد (2) مرفوعاً للأس (القوة) صفر، وتستمر قدر ما نشاء. فعلياً، يجب أن تتوقف السلسلة عند اقتراب قيم خاناتها من العدد العشري المطلوب تحويله.

نقوم بحساب قيمة كل خانة من خانات هذه السلسلة.

ننظر لقيمة الأعداد المكونة للسلسلة، ونبحث عن الخانات التي مجموع قيمها سيساوي (45). نبدأ عملية جمع قيم الخانات بدءاً من اليسار إلى اليمين.

كل خانة تم استخدام قيمتها نعطيها الرقم "1"، وكل خانة لم يتم استخدام قيمتها نعطيها الرقم "0". الآن نقوم بترتيب الواحدات والأصفار بدءاً من اليسار إلى اليمين، فنحصل على العدد الثنائي المطلوب.

هل وجدتم الكلام السابق معقداً وصعباً للفهم؟ حسناً، الصورة التالية تظهره وتوضحه كاملاً وبشكلٍ بسيط. الخانات التي نحتاجها لتشكيل العدد (45) تم إحاطتها بالمستطيل البرتقالي، بينما الخانات التي لم يتم استخدامها تم إحاطتها بالمستطيل الأزرق. كان بالإمكان وضع الخانة الموافقة للعدد (2) مرفوعاً للقوة (6) والتي تساوي القيمة (64)، ولكنها ستأخذ القيمة "0"، وقيمة "0" على اليسار تعتبر قيمةً مهملة، لذلك تم اعتماد الخانة الموافقة للعدد (2) مرفوعاً للقوة (5) كأعلى قيمة مطلوبة لتشكيل العدد (45)، بالإضافة للخانات الثنائية الأخرى.

# قوى الخانات الثنائية

| قوى الخانات | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|
| قيمة الخانة | 32 | 16 | 8 | 4 | 2 | 1 |
| السلسلة الثنائية | 1 | 0 | 1 | 1 | 0 | 1 |

$$45 = 32 + 8 + 4 + 1$$
$$(45)_{10} = (101101)_2$$

من أجل الحصول على السلسلة الثنائية الموافقة للعدد العشري، فإننا نقوم بكتابة قوى الأعداد الثنائية ونحسب قيمتها، وننظر للخانات التي يمكن عبر جمع قيمها تشكيل العدد العشري المطلوب، وكل خانة تم استخدامها نعطيها الرمز «1» بينما نعطي كل خانة لم نستخدمها الرمز «0».

بالنسبة للنظام الثماني، فإنه يمكن ترميز أي رقم بالنظام الثماني باستخدام (3) أعداد ثنائية، أما بالنسبة للنظام الستة عشري، فإنه يمكن تمثيل أي رقم فيه باستخدام (4) خانات بالنظام الثنائي. لماذا؟ لنتذكر أن العدد 2 مرفوعاً للقوة (3) هو (8)، وبالتالي، فإنه وباستخدام (3) خانات ثنائية، سيكون بالإمكان ترميز (8) أعداد، وهو المطلوب بحالة النظام الثماني. أما بالنسبة للنظام الستة عشري، فإننا يجب أن نتذكر أن 2 مرفوعاً للقوة 4 هو 16، وهذا يعني أنه باستخدام 4 خانات ثنائية نستطيع ترميز 16 قيمة، وهو المطلوب بالنسبة للنظام الستة عشري. هنا يجدر بنا الإشارة لنقطةٍ هامة، وهي أن كل عدد ثماني يحتاج لـ 3 خانات ثنائية، وهذا يعني أننا لكتابة رقم ثماني يتكون من خانتين، سنكون بحاجة لثلاث خانات لكل عدد، أي 6 خانات ككل. ولو كان العدد الثماني يتكون من 3 خانات، فإننا سنحتاج لـ 9 خانات ثنائية لترميز العدد، وهكذا. الأمر نفسه ينطبق على النظام الستة عشري، الذي يتطلب ترميز كل خانة فيه 4 خانات ثنائية، وبالتالي فإن عدداً بالنظام الستة عشري يتكون من 3 خانات مثلاً، سيتطلب 12 خانة ثنائية كي يتم ترميزه.

3 -العلاقة بين النظام الثنائي والنظامين الثماني والستة عشري

بعد أن تعرفنا على كيفية التحويل بشكلٍ مباشر بين النظامين الثنائي والعشري، علينا أن نسلط الضوء على العلاقة بين النظام الثنائي والثماني، والنظام الثنائي والنظام الستة عشري.

بالبداية، علينا أن نتذكر أن العدد (8) هو عبارة عن (3^2)، والعدد (16) هو عبارة عن (4^2)، وبالتالي، وعبر هذا المفهوم البسيط، تصبح العلاقة واضحة وسهلة:

كل عدد في النظام الثماني يمكن تمثيله عبر ثلاثة أعداد ثنائية، وكل عدد في النظام الستة العشري يمكن تمثيله عبر أربعة أعداد ثنائية.

كمثال توضيحي بسيط، فإن العدد (0) في النظام الثماني، يكتب باستخدام النظام الثنائي عبر السلسلة (000)، والعدد (0) في النظام الستة عشري، يكتب في النظام الثنائي عبر السلسلة (0000). بينما العدد (1) الثماني، فإنه يكتب ثنائياً باستخدام السلسلة (001)، والعدد (1) في النظام الستة عشري يكتب ثنائياً باستخدام السلسلة (0001). يمكن تلخيص هذه المفاهيم كاملةً باستخدام الجدول التالي البسيط.

| الأعداد الستة عشرية | الأعداد الثمانية | الأعداد الثنائية | الأعداد العشرية |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 01 | 01 | 01 | 1 |
| 02 | 02 | 10 | 2 |
| 03 | 03 | 11 | 3 |
| 04 | 04 | 100 | 4 |
| 05 | 05 | 101 | 5 |
| 06 | 06 | 110 | 6 |
| 07 | 07 | 111 | 7 |
| 08 | 10 | 1000 | 8 |
| 09 | 11 | 1001 | 9 |
| A | 12 | 1010 | 10 |
| B | 13 | 1011 | 11 |
| C | 14 | 1100 | 12 |
| D | 15 | 1101 | 13 |
| E | 16 | 1110 | 14 |
| F | 17 | 1111 | 15 |
| 10 | 20 | 10000 | 16 |
| 11 | 21 | 10001 | 17 |
| 12 | 22 | 10010 | 18 |
| 13 | 23 | 10011 | 19 |
| 14 | 24 | 10100 | 20 |
| 15 | 25 | 10101 | 21 |
| 16 | 26 | 10110 | 22 |

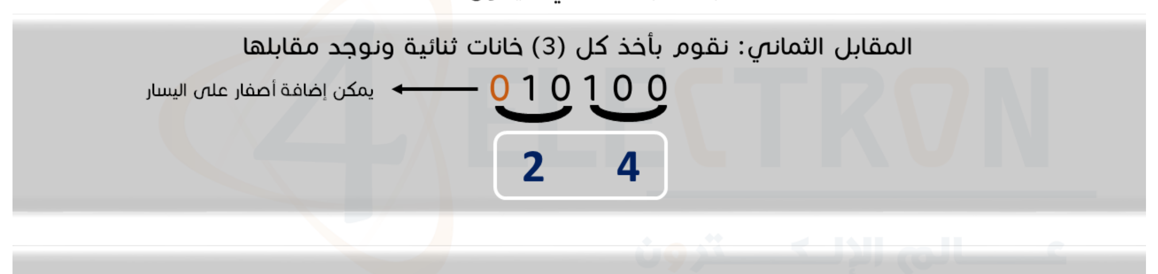
الجدول السابق يعطينا فكرة جيدة وتصور ممتاز حول أنظمة العد والعلاقة فيما بينها. فالنظام العشري هو النظام الذي نألفه ونتعامل معه، بينما النظام الثنائي هو النظام الذي يتكون فقط من عددين (0,1) ويمكن تمثيل أي عدد عبر سلسلة من الأعداد الثنائية. النظام الثنائي هو نظام أساسه العدد (8) ومكوناته هي (0,1,2,3,4,5,6,7) ونلاحظ أن العدد التالي بعد العدد (07) في النظام الثماني سيكون (11) وليس (08) مثلاً، وذلك لأن الرقم (8) ليس ضمن مكونات النظام، وكذلك الرقم (9). نفس الأمر بالنسبة للنظام الستة عشري، فأساسه العدد (16) ومكوناته هي من (0) وحتى (15)، وللتفريق بين أعداده وأعداد النظام العشري، تم ترميز الأعداد من (10) وحتى (15) بالأحرف (A,B,C,D,E,F)على التوالي. نلاحظ أن العدد التالي بعد العدد (15) في النظام الستة عشري هو (11)، أي بداية تعداد جديد، وليس العدد (16)، لأن العدد 16 ليس ضمن مكونات النظام. كنتيجة، فإن كل نظام أعداد يقوم بتكرار نفسه بعد انتهاء مكوناته، ولكننا ننتقل لخانةٍ جديدة. فبعد انتهاء أول ثمانية أعداد من النظام الثماني، فإننا نقوم بتكرارها، ولكن ضمن خانة (1)، بدلاً من خانة (0) التي تم البدء بها، وعند انتهاء المكونات الثمانية المرتبطة الرقم (1) يعيد النظام نفسه ولكن مع خانة (2) وهكذا، ونفس الأمر ينطبق تماماً على النظام الستة عشري.

الأمر الثاني الهام الذي نلاحظه من الجدول، هو أن كل رقم ثماني يمكن ترميزه باستخدام (3) أعداد ثنائية، وكل رقم ستة عشري يمكن ترميزه باستخدام (4) خانات ثنائية. فالعدد الثماني (10) مثلاً، يقابل بالجدول العدد الثنائي (1000)، ولو نظرنا له، فالعدد (10) مكون من رقمين: 0 و 1. العدد 0 يأخذ الخانات الثنائية الثلاثة الأولى من اليمين لليسار وهي (000)، والعدد 1 يأخذ الخانات الباقية. طبعاً، وكي يكون العدد (10) مرمزاً بشكل صحيح بالنظام الثنائي، فإنه يجب أن يتكون من (6) خانات، والأمر هنا بسيط جداً، فكل ما علينا القيام به هو إضافة عدد مناسب من الأصفار على اليسار، حتى يصبح العدد الثماني مرمزاً بشكل صحيح ثنائياً. كنتيجة، فإن المقابل الثماني للعدد (10)، هو (001000). الخانات الثلاث الأولى من اليمين (000) تقابل العدد (0) الثماني، والخانات الثلاث التالية (001) تقابل العدد (1). هذا الأمر ينطبق على كل الأعداد الثمانية. بالنسبة للأعداد الستة عشرية، فإن الطريقة نفسها تنطبق، ولكن على أربعة خانات: أي أن كل عدد ستة عشري يجب أن يتكون من أربعة خانات ثنائية.

يمكن فهم العلاقات السابقة باستخدام المثال التالي: سنأخذ العدد العشري (20)، وسنوجد مقابله الثنائي، والثماني، والستة عشري، وذلك عبر الصورة التوضيحية التالية:

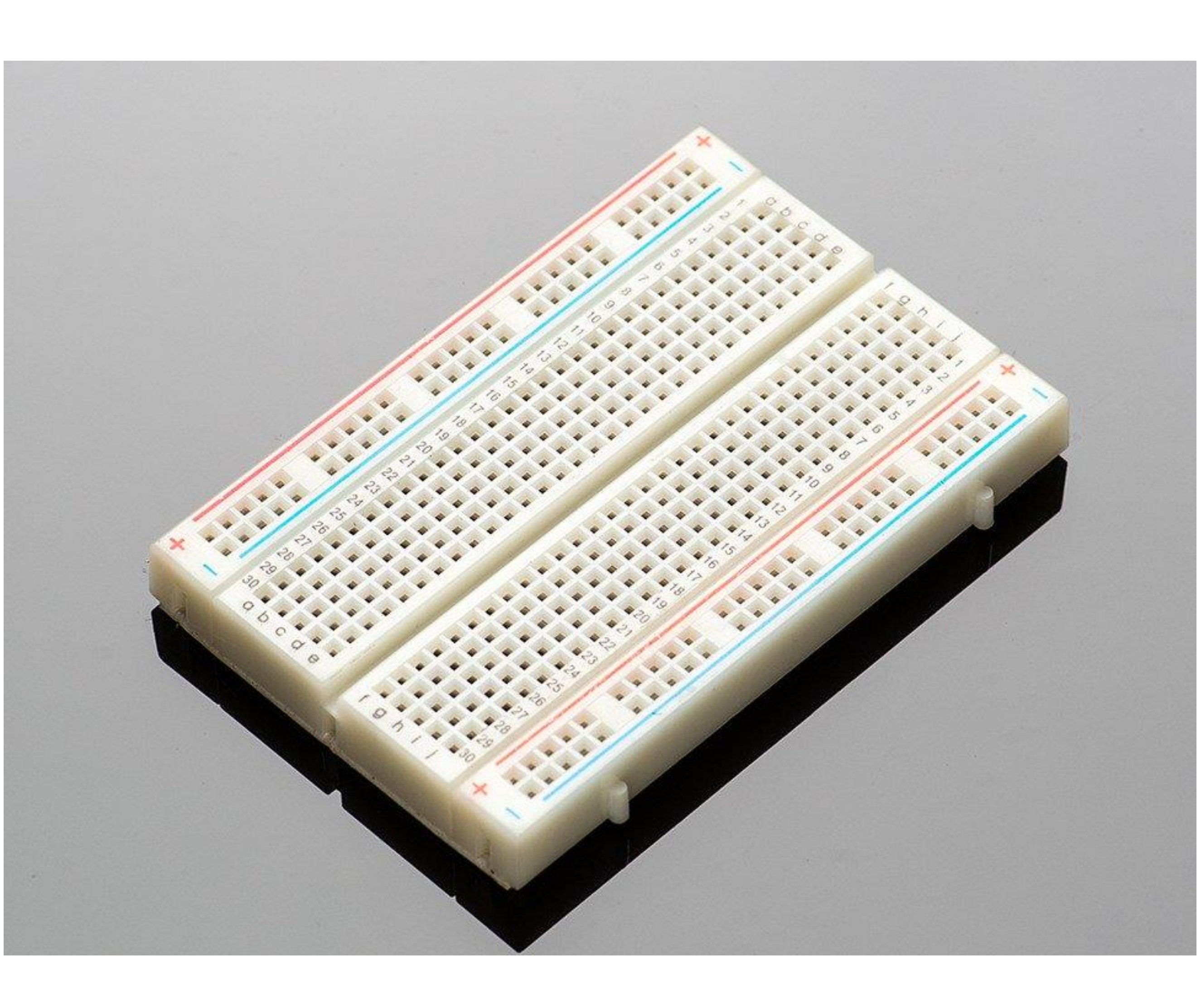
التحويل بين أنظمة العد

ليكن لدينا العدد العشري 20
نحسب مقابله الثنائي فيكون: 10100




إلى هنا، نكون قد غطينا كافة المواضيع الأساسية المتعلقة بأنظمة العد، وأسسها، ومبادئ التحويل فيما بينها. هنالك مواضيع وقضايا إضافية تتعلق بالأعداد ذات الفاصلة وكيفية تمثيلها، أو إجراء العمليات الحسابية على أنظمة العد سوف ندرسها لاحقا في بعض المواضيع.

# لوح التوصيل *Breadboard*



تُعتبر ألواح التجارب أحد أهم المكُونات الأساسية لتعلم كيفية بناء الدوائر الكهربائية في هذا الدرس ستتعلم ماهي ألواح التجارب، وكيفية استخدام لوح التجارب. بعد الانتهاء من المُفترض أن يكون لديك الفهم الأساسي لكيفية عمل ألواح التجارب وأن تكون قادراً على بناء الدوائر على لوح التجارب.

لماذا نستخدم ألواح التجارب؟

ألواح التجارب لا تتطلب استخدام اللحام لتثبيت المكونات على الإطلاق. إنها مُكونات رائعة لصنع الدائر المؤقتة، وعمل النماذج الأولية.

صناعة النماذج الأولية (prototyping) هي العملية التي يجري فيها اختبار فكرة ما عن طريق صناعة نموذج أولي، والذي منه يتم تطوير ونسخ الأشكال الأخرى، وهذا أحد الاستخدامات الرئيسية لألواح التجارب. إذا لم تكن متأكداً من طريقة تفاعل الدائرة الخاصة بك تحت مجموعة من العوامل فأفضل ما تفعله هو نموذج أولي لتجربة الدائرة واختبارها.

أما بالنسبة لحديثي العهد بالإلكترونيات والدوائر الكهربية فألواح التجارب هي أفضل شيء يمكنهم البدء به. أفضل ما يميز ألواح التجارب هي قدرتها على استيعاب الدوائر البسيطة بالإضافة للدوائر المعقدة للغاية. سترى لاحقاً في هذا الدرس أنه إذا زاد حجم دائرتك للغاية ولم يعد لوح التجارب خاصتك يكفيها، يمكنك توصيل أي عدد من ألواح التجارب معاً لكي تتسع للدوائر بجميع أحجامها وتعقيداتها.

أحد الاستخدامات الأخرى الشائعة لألواح التجارب هو اختبار الأجزاء الجديدة، مثل الدوائر المتكاملة((Integrated circuits (ICs) ، عندما تريد معرفة كيف يعمل شيء ما وبحاجة لتغيير توصيل الأسلاك بشكل مستمر لا يجب عليك لحام الوصلات والأسلاك باستمرار، ببساطة استخدم لوح تجارب.

كما ذكرنا سلفاً لسنا نريد دائماً أن نصنع دوائر دائمة. على سبيل المثال، في SparkFun عادة ما يستخدم فريق الدعم الفني ألواح التجارب لاستنساخ دائرة العميل التي تحتوي على مشكلة، حيث يتم بناء الدائرة على لوح تجارب ثم اختبارها وتحليلها. يمكن ببساطة توصيل المكوّنات الخاصة بالعميل، وبعد تكوين الدائرة واكتشاف المشكلة، يتم تفكيك كل شيء وحفظه جانباً للاستخدام لتشخيص مشاكل في دوائر أخرى.

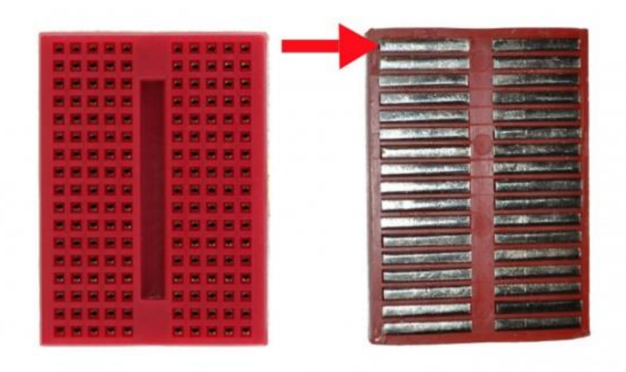
تركيب لوح التجارب




أفضل طريقة لشرح طريقة عمل ألواح التجارب هي تفكيكها لرؤية ما بدخلها. يمكننا استخدام لوح تجارب صغير الحجم لتسهيل ملاحظة طريقة عمله لنرى ما يلي :

### الأشرطة الطرفية (terminal strips)

هذا ما يبدو عليه لوح التجارب بعد إزالة غطاءه. يمكنك رؤية العديد من الصفوف الأفقية مصنوعة من شرائط معدنية بأسفل لوح التجارب.



على اليسار لوح تجارب، وعلى اليمين نفس لوح التجارب بعد أن قمنا بقلبه وإزالة غطاءه

الجزء العلوي من تلك الصفوف المعدنية يحتوي على مشابك (clips) صغيرة تختفي تحت الفتحات البلاستيكية الموجودة بلوح التجارب. هذه المشابك تتيح لنا وضع سلك أو طرف أي مكون بداخل فتحات اللوح، ومن ثوم تقوم تلك المشابك بتثبيتها في مكانها.

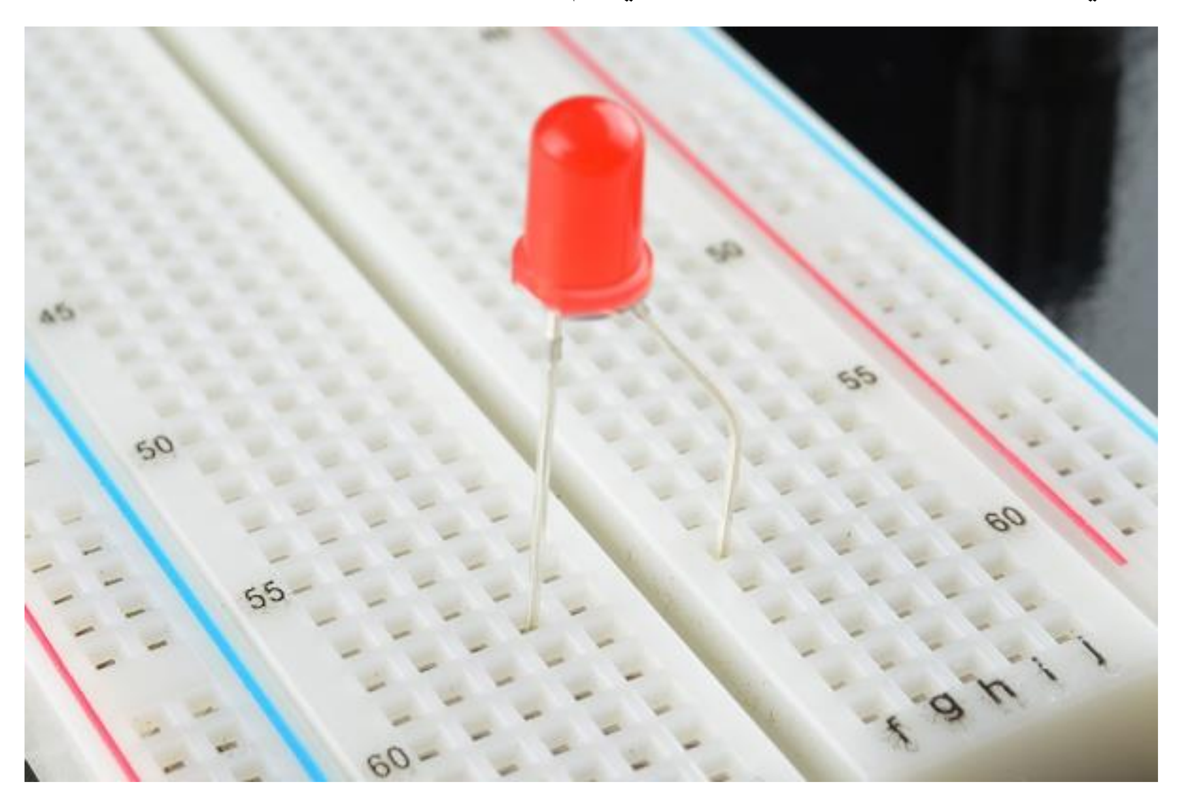
شريط معدني موصل للكهرباء تمت إزالته من لوح التجارب الموجود في الأعلى:



عند وضع أي قطعة في إحدى الفتحات فإنها تصبح متصلة كهربياً بأي شيء آخر موضوع في نفس الصف. هذا لأن هذه الأشرطة المعدنية موصلة للكهرباء وتسمح بسريان الكهرباء بين جميع النقاط لموجودة على نفس الصف.

لاحظ أنه يوجد فقط خمسة مشابك في هذا الشريط. هذا هو المُتبع في معظم ألواح التجارب. لذلك يمكنك توصيل حتى خمسة مكونات فقط في أي شريط، لماذا لا يمكن توصيل أكثر من خمسة مكونات؟ نلاحظ ان كل صف أفقي يقطعه أخدود أو شق في المنتصف من لوح التجارب. هذا الشق يعزل كلا الجانبين من أي صف عن بعضهما البعض، وبالتالي لا يصبح الجانبان متصلين كهربياً. سنشرح الغرض من هذا الأخدود فيما بعد، لكن الآن تذكر فقط أن كلا الجانبين من أي صف يكونان غير متصلين مع بعضهما البعض، مما يجعلنا نحصل على خمس نقاط لتثبيت المكونات على كلا الجانبين.

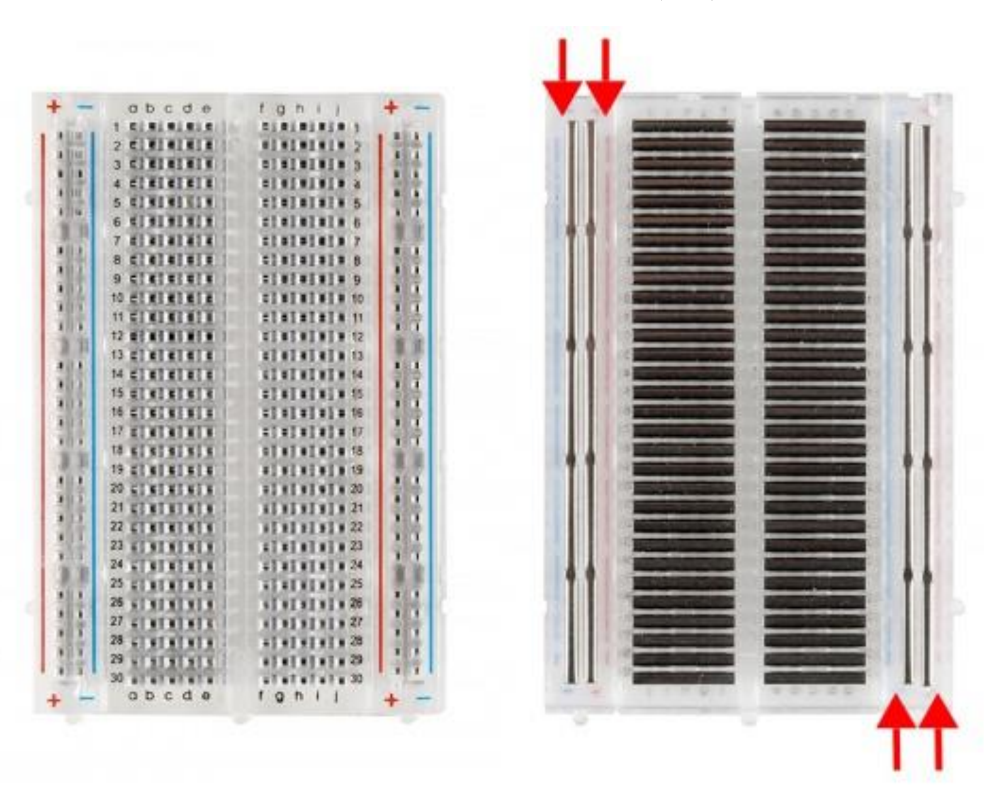
ديود ضوئي مثبت على لوح تجارب. لاحظ أن كل طرف من الديود موجود على أحد جانبي الشق. وهذا لحماية الوصلات التي يتم توصيلها بالديود من القصر.



## سكك التغذية (power rails)

الآن أصبحنا على علم بكيفية صناعة الوصلات في لوح التجارب، دعونا نرى لوحاً أكبر. تحتوي ألواح التجارب –بجانب الصفوف الأفقية- على ما نطلق عليه سكك التغذية التي توجد بشكل عمودي بطول جانبي اللوح.

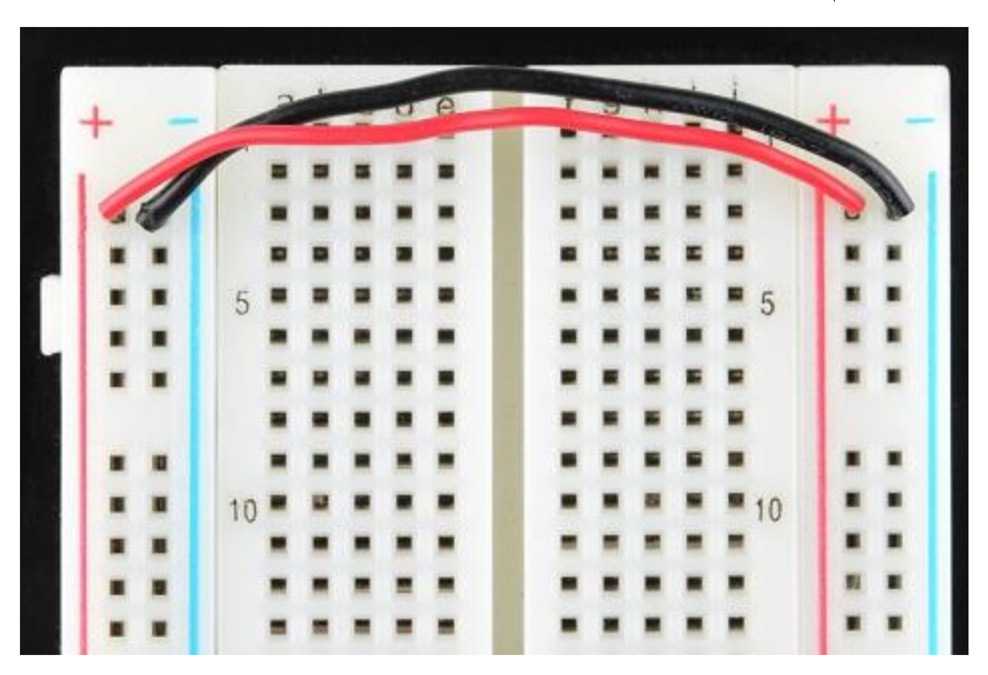
لوح تجارب متوسط الحجم، تم إزالة الغطاء الخاص به لرؤية سكك التغذية:



سكك التغذية هي عبارة عن شرائط معدنية عمودية مماثلة للأشرطة الطرفية الأفقية، فيما عدا أنها عادة* ما تكون متصلة بشكل كامل. عند بناء الدوائر الكهربية تحتاج لتوصيل الطاقة للعديد من الأماكن المختلفة. سكك التغذية تمنحك سهولة توصيل الطاقة الكهربية لأي نقطة في الدائرة. عادة ما يتم تمييز سكك التغذية بإشارات (+) أو (-) أو باستخدام خطوط ملونة بالأحمر أو الأزرق أو الأسود لتمييز الجانب الموجب من الجانب السالب.

يجب أن تنتبه أن سكك التغذية على جانبي لوح التجارب غير متصلة، لذلك إذا أردت أن تقوم بتوصيل نفس مصدر الطاقة بسكك التغذية على كلا الجانبين فيجب أن تقوم بتوصيل الجانبين باستخدام بعض أسلاك القفز (jumper wires).

تذكر أن العلامات والألوان الموجودة على لوح التجارب ليست إلزامية وإنما مرجعية فقط. ليست هناك قاعدة تُلزمك بتوصيل الطرف عالي الجهد بسكة التغذية الموجبة (+) أو ان تُوصِّل الأرضي (ground) بسكة التغذية السالبة (-)، ولكنها وسيلة جيدة يمكنك استخدامها للتنظيم.

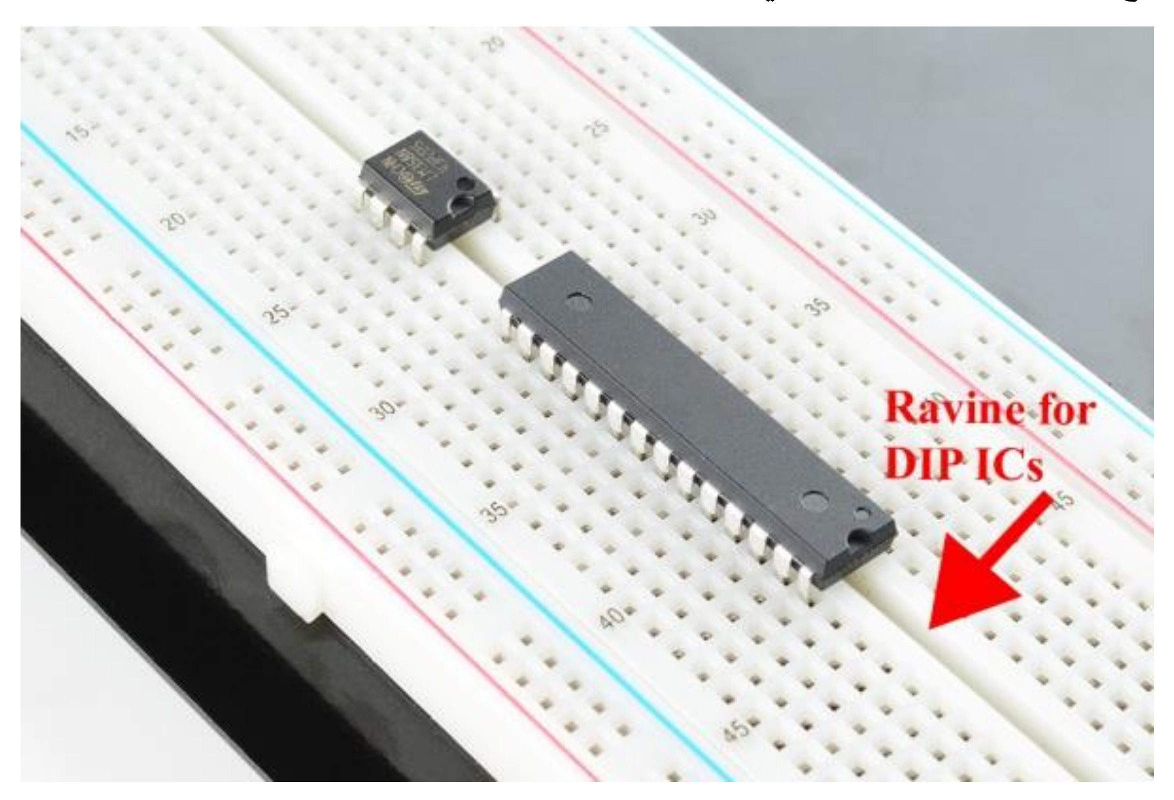



سلكا قفز يتم استخدامهما لتوصيل سكك التغذية على كلا الجانبين ببعضها البعض. يجب أن تقوم بتوصيل الطرفين الموجبين ببعضهما البعض وكذلك الطرفين السالبين.

## نقاط الدعم الثنائي (DIP support)

ذكرنا فيما سبق أن هناك شق يفصل جانبي لوح التجارب. هذا الشق له أيضاً استخدام هام للغاية؛ العديد من الدوائر المتكاملة (integrated circuits) التي عادة ما يطلق عليها ICs أو الرقاقات (chips) يتم تصنيعها بشكل خاص لتتناسب مع ألواح التجارب. من اجل تقليل المساحة التي تشغلها على لوح التجارب فإنها تأتي في شكل يُطلق عليه حزمة خطية ثنائية ( Dual in-line Package (]ual in-line Package ()علي عليه ى لوح التجارب فإنها تأتي على شكل ما يطلق عليه بتوصيل نفس مصدر الطاقة بسكك التغذية على كلا الجانDIP)).

رقائق DIP يكون لها سيقان في كلا الجانبين تتناسب تماماً مع الشق الموجود في المنتصف. لأن كل ساق من سيقان الدائرة المتكاملة تكون فريدة ومستقلة فإننا لا نريد أن يتم توصيل السيقان من كلا الجانبين معاً، هنا تبرز أهمية الشق الموجود في منتصف لوح التجارب. لذلك يمكننا توصيل المكونات على جانبي الدائرة المتكاملة بدون التداخل مع وظيفة الساق الموجودة في الطرف الآخر.




دائرتان متكاملتان من نوع DIP. في الأعلى LM358 مضخم عملياتي شائع، وفي الأسفل متحكم دقيق ATMega328.

**الصفوف والأعمدة**

ربما تكون قد لاحظت أن العديد من ألواح التجارب يكون بها أرقام وحروف موضوعة على العديد م الصفوف والأعمدة. هذه الأرقام والحروف ليس لها أي وظيفة أو غرض سوى مساعدتك عند بناء دائرتك. الدوائر الكهربية يمكن أن تصبح مُعقدة سريعاً، ومُجرد وضع سلك واحد أو ساق واحدة لأحد المكونات في مكان خاطئ كفيل بجعل الدائرة بأكملها لا تؤدي وظيفتها أو لا تعمل على الإطلاق. إذا كنت تعرف رقم الصف الذي تقوم بالتوصيل فيه فهذا سيجعل توصيل الأسلاك أكثر سهولة وبساطة بدلاً من التحديق بعينك لتحديد الصف في كل مرة.

هذه الأرقام والحروف مُفيدة أيضاً عند استخدام كُتيبات التعليمات (instruction booklets).الكثير من الكتب والأدلة تحتوي على مُخططات الدوائر كهربية لكي تقوم باتّباعها أثناء بناء دوائرك الكهربية.

تذكر أن الدائرة الكهربية التي تقوم ببنائها لا يلزم أن تكون في نفس الموقع من لوح التجارب كما هي في الكتاب. في الواقع ليس من الضروري أن تبدو مشابهة لها من الأساس. يمكنك بناء الدائرة بأي شكل تريد طالما أنك تقوم بعمل التوصيلات بشكل سليم.

عندما تقوم ببناء دائرة كهربية لست ملزماً باستخدام لوح تجارب واحد فقط، بعض الدوائر تتطلب استخدام عدد من الألواح لاحتياجها لمساحة كبيرة نتيجة لكثرة مكوناتها وتوصيلاتها. العديد من ألواح التجارب تحتوي على بروزات وفتحات على جوانبها، وبعضها يحتوي على المزيد منها في الجزء العلوي والسفلي. هذا يُتيح لك توصيل أي عدد من ألواح التجارب معاً للحصول على مساحة لا نهائية لعمل نماذجك ودوائرك.



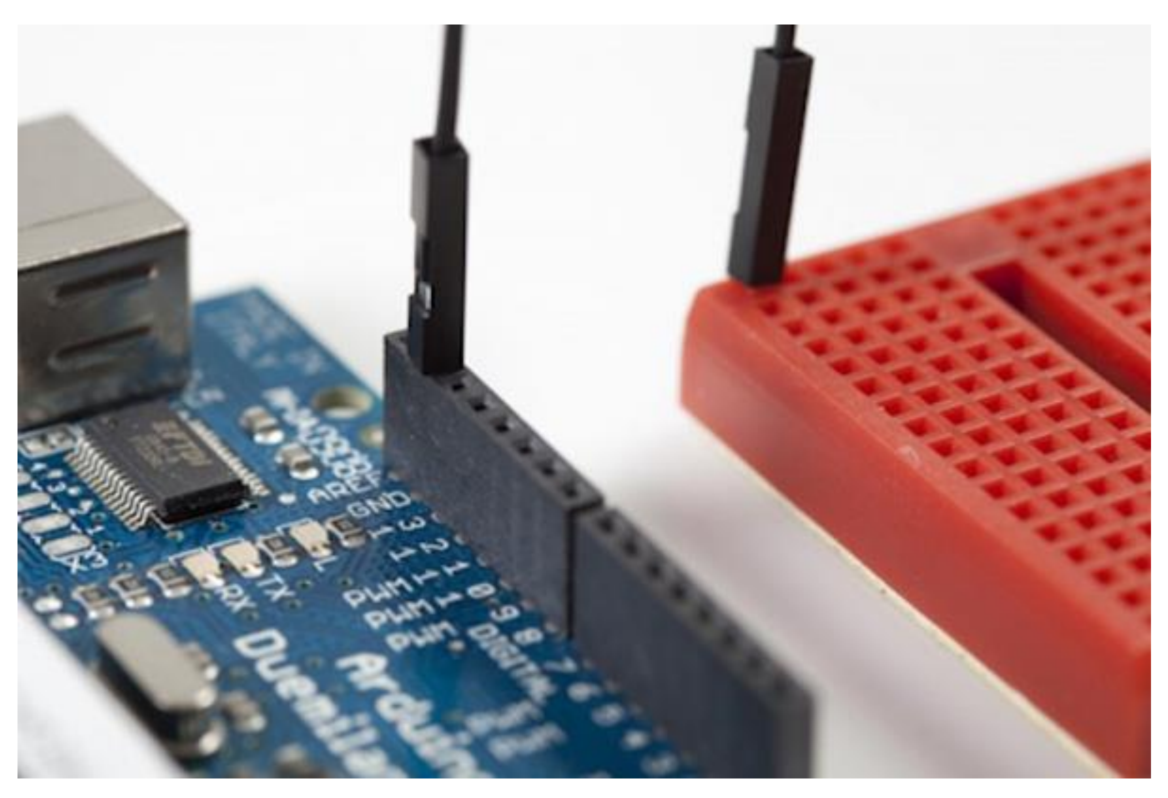
الشكل اعلاه أربعة ألواح تجارب متصلة معاً , هناك أيضاً بعض ألواح التجارب تحتوي على غطاء يسمح بتثبيتها على العديد من الأسطح المختلفة. هذه خاصية مفيدة إذا كنت ترغب بتثبيت لوح التجارب بداخل غلاف ما أو دمجه مع شيء آخر.

## توصيل الطاقة إلى لوح التجارب

هناك العديد من الخيارات المتاحة لتغذية لوح التجارب بالطاقة.

### التوصيل من مصدر طاقة آخر

إذا كنت تستخدم لوحاً تطويرياً مثل أردوينو (Arduino) يمكنك ببساطة توصيل لوح التجارب بمنافذ الطاقة الخاصة به. يحتوي الأردوينو على العديد من منافذ الطاقة والأرضي تستطيع توصيلها بسكك التغذية أو أي من الصفوف الأخرى الموجودة في لوح التجارب.




توصيل منفذ الأرضي (GND) من بطاقة أردوينو إلى أحد صفوف لوح تجارب صغير. كل سلك يتصل بهذا الصف يكون متصلاً بالأرضي

يستمد الأردوينو الطاقة عادة من الحاسب عن طريق منفذ USB، أو من أي مُزود طاقة خارجي مثل حزم البطاريات، أو محول التيار المتردد الذي يتم توصيله بمأخذ الحائط.

## توصيل الكهرباء عن طريق قطع الربط

كما ذكرنا سابقاً في هذا الدرس، بعض ألواح التجارب تحتوي على قطع ربط تتيح لك توصيلها بمصادر الطاقة الخارجية.

الخطوة الأولى لتزويد لوح التجارب بالطاقة عن طريق قطع الربط هو توصيلها باستخدام أسلاك قفز (قد يبدو لك أن قطع الربط مُتصلة أساساً بلوح التجارب ولكن هذا غير صحيح). فكما رأينا يمكنك تخصيص ألواح التجارب بشكل كامل كما تريد، ولا تختلف قطع الربط في هذا.

علينا إذن توصيل أسلاك بقطع الربط لتوصيلها بلوح التجارب. لعمل ذلك يتم فك قطع الربط حتى ينكشف الثقب الموجود فيها. ثم يتم إدخال الجزء المُقشر (العاري) من السلك بداخل الثقب، ثم يتم ربط القطع مرة اخرى حتى تصبح الأسلاك متصلة بشكل سليم.



كل ما عليك فعله هو توصيل سلك الطاقة وسلك الأرضي بقطع الربط الخاصة باللوح. في حالة الحاجة لاستخدام تيار متردد يمكنك استخدام قطعة الربط الثالثة.

الآن أصبحت قطع الربط مُتصلة بلوح التجارب، ولكننا لم نوصل مصدر للطاقة بعد. يمكنك استخدام العديد من الطرق لتوصيل الطاقة إلى قطع الربط، ومنها إلى لوح التجارب.

## مُزودات الطاقة (Benchtop Power Supplies)

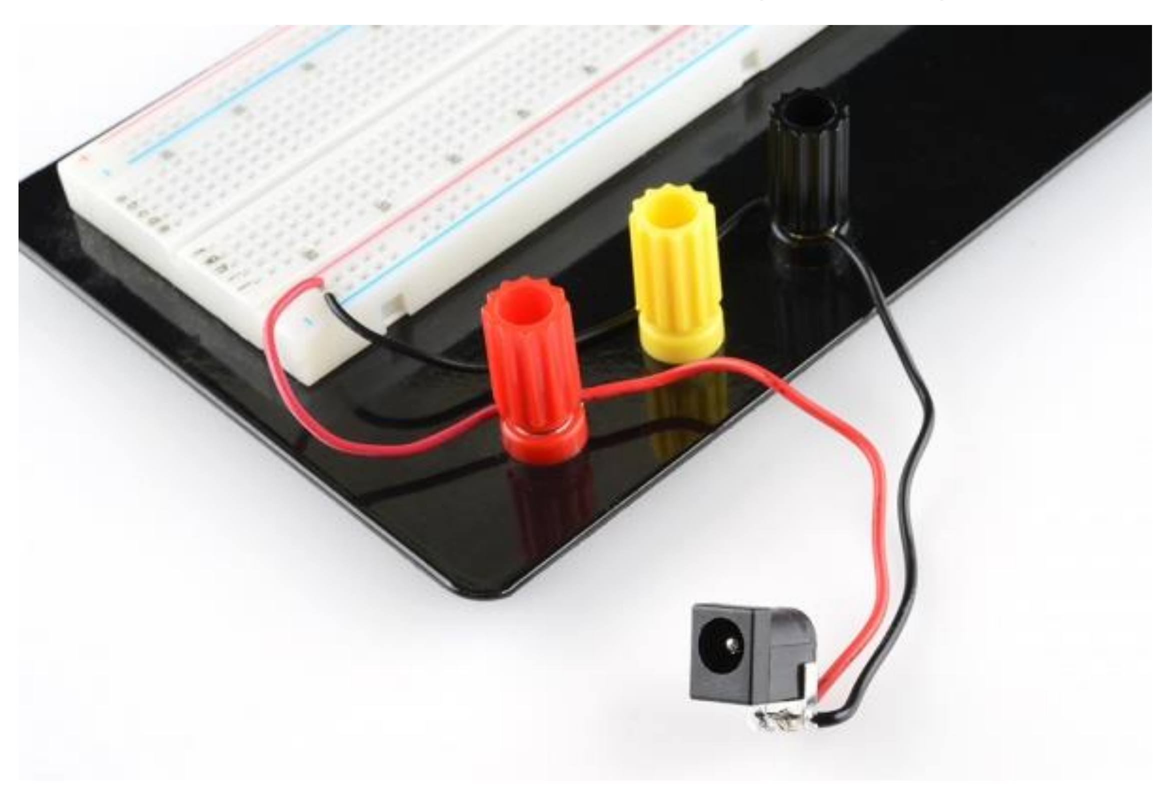
تحتوي العديد من المعامل الإلكترونية على مُزودات طاقة تتيح لك الحصول على مدى واسع من قيم الجهد والتيار لإمداد دائرتك الكهربية بالطاقة. باستخدام وصلة موزة (banana connector) يمكن توصيل الطاقة من مُزود الطاقة إلى قطع الربط.

الشكل التالي لوح تجارب يتم تزويده بالطاقة عن طريق قطع التوصيل باستخدام وصلات banana :



يمكنك أيضاً استخدام كابلات من نوع IC hooks أو alligator clips أو أي كابل آخر مع وصلة banana لتوصيل لوح التجارب بأنواع مُزودات الطاقة المختلفة.

أحد الطرق الأخرى لاستخدام قطع الربط هي لحام قابس أسطواني (barrel jack) مع بعض الأسلاك ثم توصيلها بقطع الربط. هذه الطريقة متقدمة، وتتطلب بعض مهارات اللحام (soldering).

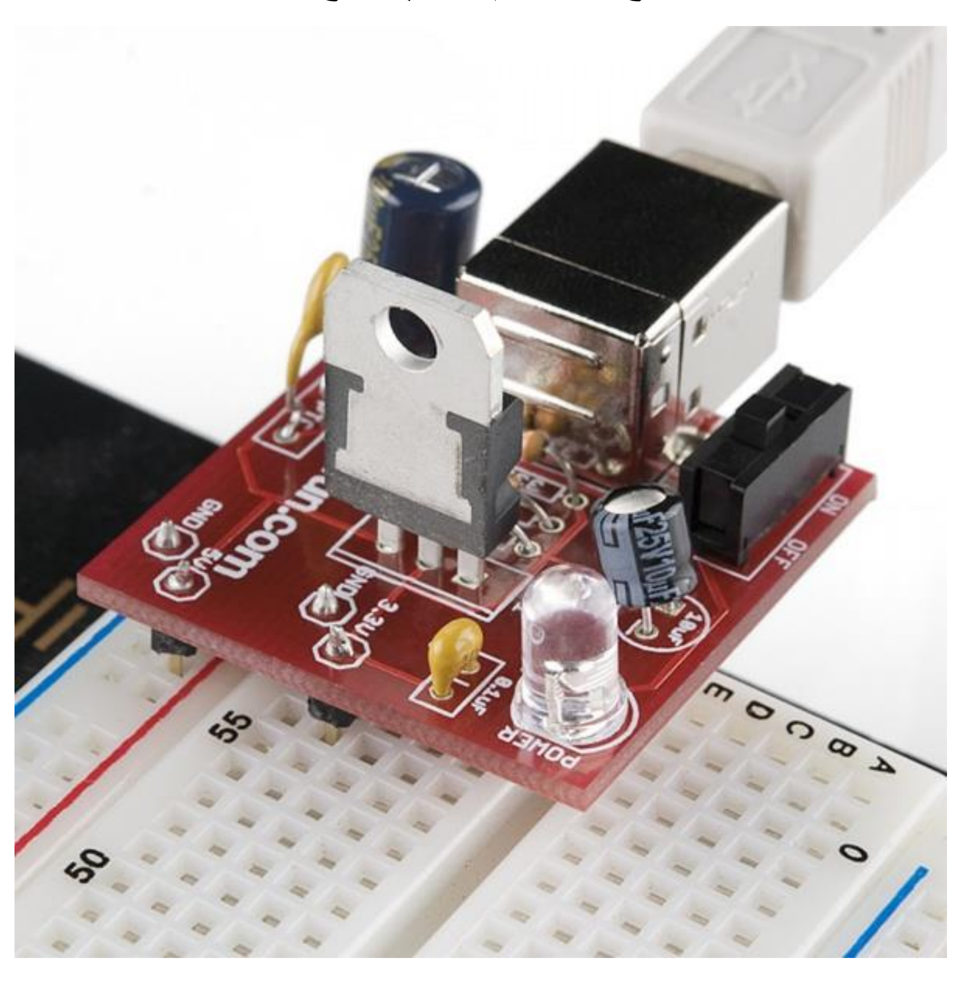


القابس الاسطواني تم لحامه مع سلكين يتصلان بقطع الربط ومنها إلى لوح التجارب. إذا لم يكن لوح التجارب خاصتك يحتوي على قطع ربط يمكنك توصيل الأسلاك من القابس الاسطواني مباشرة إلى سكك التغذية في لوح التجارب.

## مُزودات الطاقة الخاصة بألواح التجارب (Breadboard Power Supplies)

أحد الطرق الأخرى لتوصيل الطاقة لألواح التجارب هي استخدام أحد مُزودات الطاقة المتوفرة الخاصة بألواح التجارب. هناك العديد من البطاقات والمستلزمات يمكنك استخدامها لتوصيل الطاقة مباشرة للوح التجارب الخاص بك (انظر على سبيل المثال). بعضها يسمح لك استخدام محول تيار متردد وتوصيله مباشرة باللّوح. البعض الآخر يسمح لك بتوصيل الطاقة مباشرة من الحاسب الخاص بك من خلال وصلات USB. جميع تلك الوسائل تتيح إمكانية تعديل الجهد وتتيح لك جميع قيم الجهد الشائع استخدامها مع الدوائر الإلكترونية.

الشكل التالي مُزود طاقة خاص بألواح التجارب يعمل باستخدام وصلة USB تستمد الطاقة من جهاز كمبيوتر، وتتيح للمستخدم جهدي خرج هما 3.3 فولت أو 5 فولت.



## بناء دائرتك الأولى على لوح تجارب!

أصبحت لدينا الآن معرفة مناسبة بتركيب ومكونات ألواح التجارب وكيفية تغذيتها بالطاقة، كيف نستخدمها؟ سنبدأ في تفصيل ذلك بدائرة بسيطة.

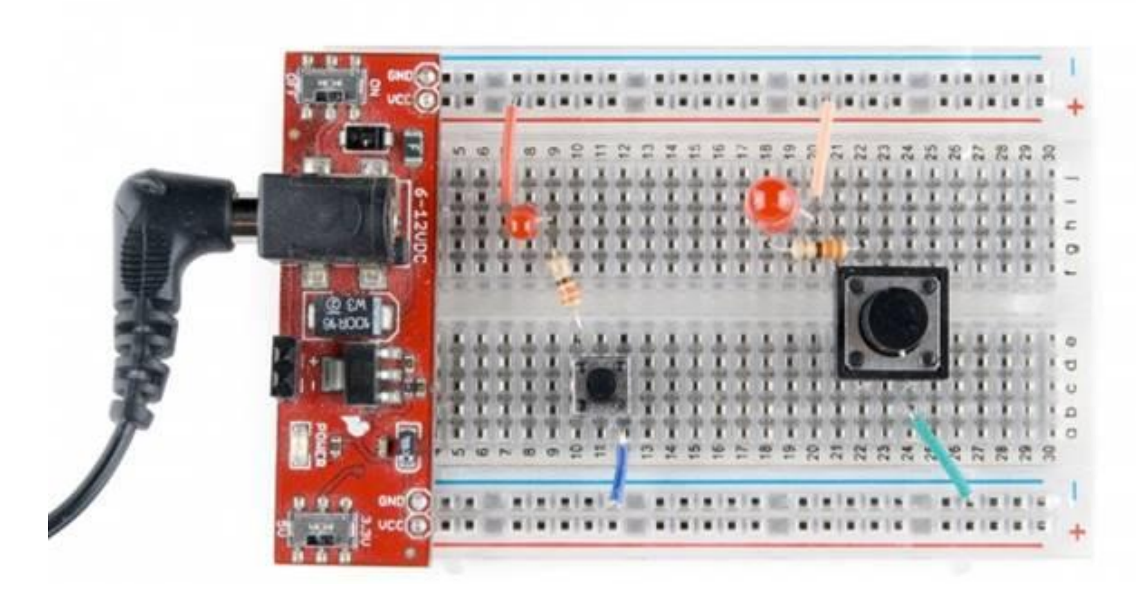
المكونات التي تحتاجها

| | |
|---|---|
| لوح تجارب | |
| مزود طاقة خاص بألواح التجارب | |
| مُحول تيار متردد تيار إلى مستمر لتوصيل الكهرباء إلى مزود الطاقة | |

| | |
|---|---|
| | ديود ضوئي (LED) |
| | مقاوم (resistor) بقيمة 330 أوم وقدرة 6/1 وات من النوع الذي يتم تثبيته عبر الثقوب |
| | سلك توصيل |
| | مفتاح ضغط ( push button (switch |

بناء الدائرة

الصورة التالية للدائرة بعد بناءها على لوح التجارب




دائرة بسيطة تحتوي على زر (button) وديود ضوئي ومُقاوم، مبنية بطريقتين مختلفتين

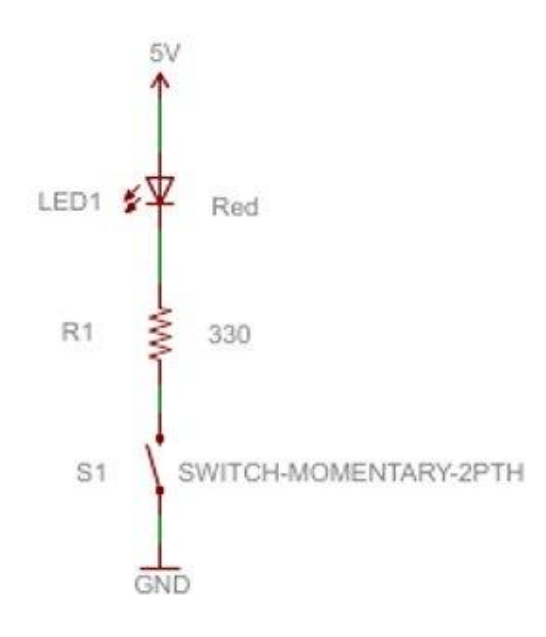
اللوح الأحمر الذي تراه في الصورة هو مزود طاقة خاص بألواح التجارب، يتم توصيله بمحول تيار متردد/ مستمر بقيمة 9 فولت، وينتج عن هذا المزود جهد 5 فولت يتم توصيله إلى سكك التغذية.

**تركيب الدائرة:**

- يتم توصيل سكة التغذية V5 بالطرف الموجب من الديود الضوئي (المصعد) باستخدام سلك التوصيل.
- يتم توصيل الطرف السالب من الديود الضوئي (المهبط) مع مقاوم بقيمة Ω330.
- بعد ذلك يتم توصيل المقاوم بالزر.
- عندما يتم ضغط الزر يتم توصيل الدائرة بالأرضي وتصبح مكتملة وبالتالي يضيء الديود الضوئي.

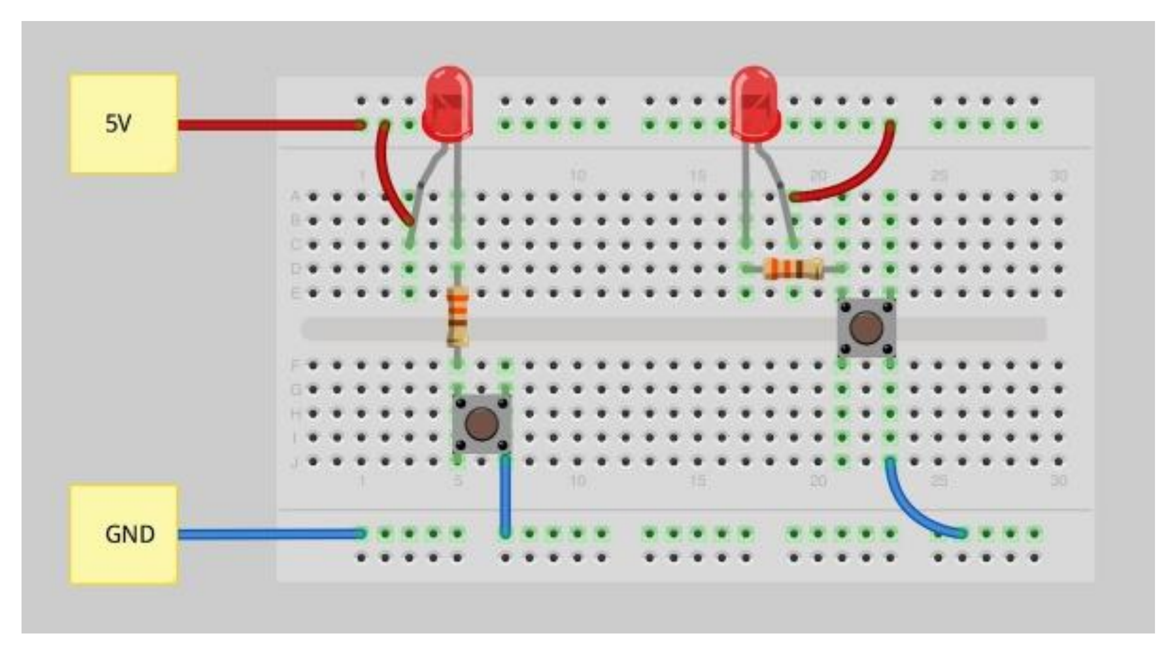
## مخططات الدوائر الكهربية (Circuit Schematics)

الصورة التالية توضح الرسم التخطيطي الخاص بالدائرة التي قمنا بتكوينها في الأعلى. مصدر الطاقة V5 ممثل بسهم في الأعلى. بعد ذلك تسري الكهرباء خلال الديود الضوئي LED (يُرمز له بمثلث به خط في المنتصف وسهمان خارجان منه). الديود يتصل بمقاوم (الخط المتعرج) والذي بدوره يتصل بالزر (رمزه يشبه المزلاج). في النهاية يتم توصيل الزر بالأرضي (الخط الأفقي في الأسفل).




ربما تبدو هذه الطريقة لرسم الدوائر الكهربية مضحكة، ولكنها طريقة هامة وأساسية تم استخدامها لعقود طويلة. تلك المخططات تسمح لأشخاص من جنسيات مختلفة ويتكلمون بلغات مختلفة، تسمح لهم ببناء والتعامل مع الدوائر الكهربية المُصممة بواسطة أي شخص. كما ذكرنا سلفاً، يمكنك بناء دائرة واحدة بعدة طرق. لكن -كما نرى في الرسم التخطيطي- لا بد من عمل توصيلات معينة، وعدم اتباع هذا الرسم التخطيطي سيعطينا دائرة مختلفة تماماً.

آخر شيء يلزمك معرفته هو أن هناك العديد والعديد من المصادر والبرامج التي يمكنك استخدامها لبناء الدوائر الكهربية بدون استخدام لوح تجارب حقيقي. أحد أكثر تلك البرامج شيوعاً هو Fritzing. هذا البرنامج هو برنامج مجاني يتيح لك بناء الدوائر الكهربية على ألواح تجارب افتراضية. وكذلك يعطيك رسوم تخطيطية للدوائر التي تقوم ببنائها. هذا الشكل يوضح الدائرة التي قمنا ببنائها على برنامج Fritzing:




لاحظ أن الخطوط الخضراء توضح طريقة اتصال المكونات المختلفة في الدائرة

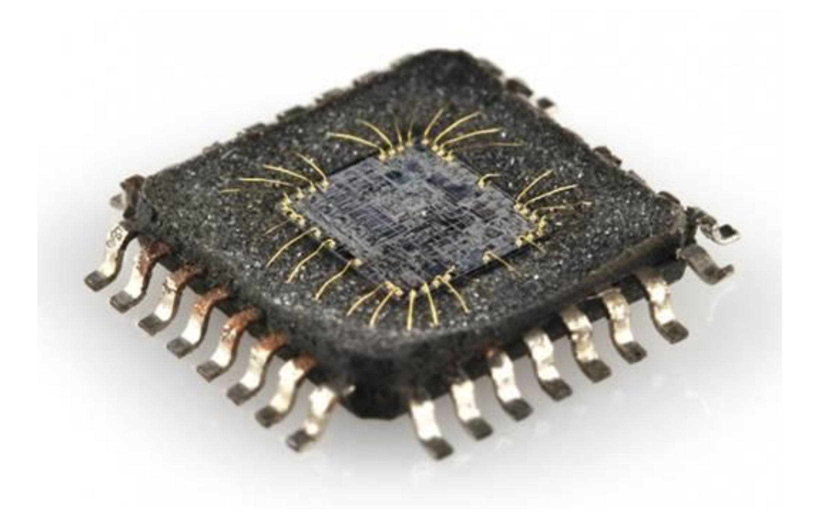
هناك العديد من البرامج المشابهة لـ Fritzing. بعضها مجاني، والبعض الآخر مدفوع. بعضها يتيح أيضا بناء الدوائر واختبار أداءها عن طريق المحاكاة. قم بالبحث في الانترنت واعثر على أفضل أداة تناسبك.

# الدوائر المتكاملة

تعتبر الدوائر المتكاملة (ICs) عماد وأساس الإلكترونيات الحديثة، فهي المكون الأهم والأكثر فعالية في معظم الدوائر الإلكترونية. إنها الشرائح السوداء الصغيرة التي تراها حولك في كل مكان وفي كل لوحة إلكترونية على الاطلاق , انها العالم **!**. إذا لم تكن شخصاً مولعاً بالإلكترونيات التناظرية فبالتأكيد يحتوي كل مشروع من المشاريع التي تبنيها على دائرة متكاملة واحدة على الأقل؛ لذلك من المهم للغاية أن نفهم ما هي الدوائر المتكاملة من الداخل ومن الخارج.

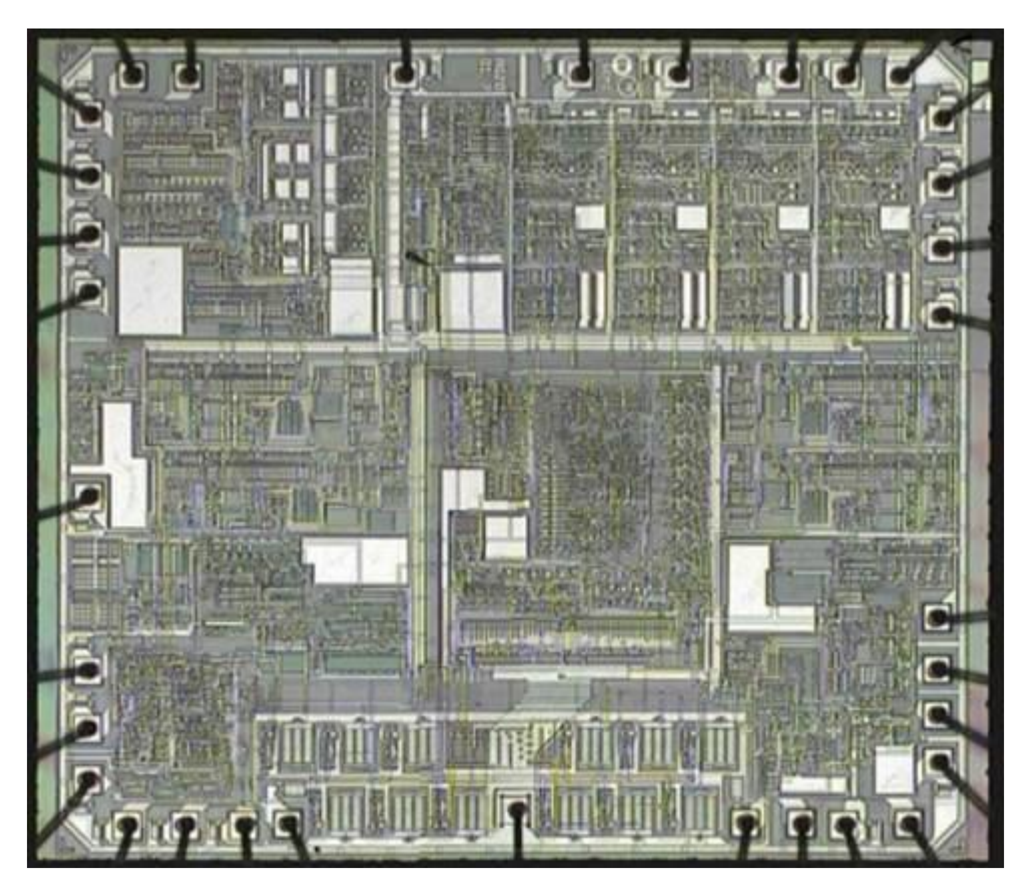
تحدثنا سابقا ان الدائرة المتكاملة هي عبارة عن مجموعة من المكونات الإلكترونية (مقاومات(resistors) ، ترانزستورات(transistors) ، مكثفات(capacitors)، الخ) مرصوصة معاً على شريحة دقيقة، وموصلة معاً لتحقيق هدف مشترك. تأتي الدوائر المتكاملة بأشكل مختلفة: بوابات منطقية أحادية الدائرة (single-circuit logic gates)، مضخمات عملياتية(op amps) ، مؤقتات 555 timers، منظمات جهد(voltage regulators) ، أجهزة التحكم في المحركات (motor controllers)، المتحكمات الدقيقة(microcontrollers) ، المعالجات الدقيقة (microprocessors)، ومصفوفات البوابات المنطقية القابلة للبرمجة (FPGAs) والقائمة تطول ولا يسعنا ذكر جميع التطبيقات هنا.

يوجد داخل الدوائر المتكاملة بعد إزالة غطاءها :

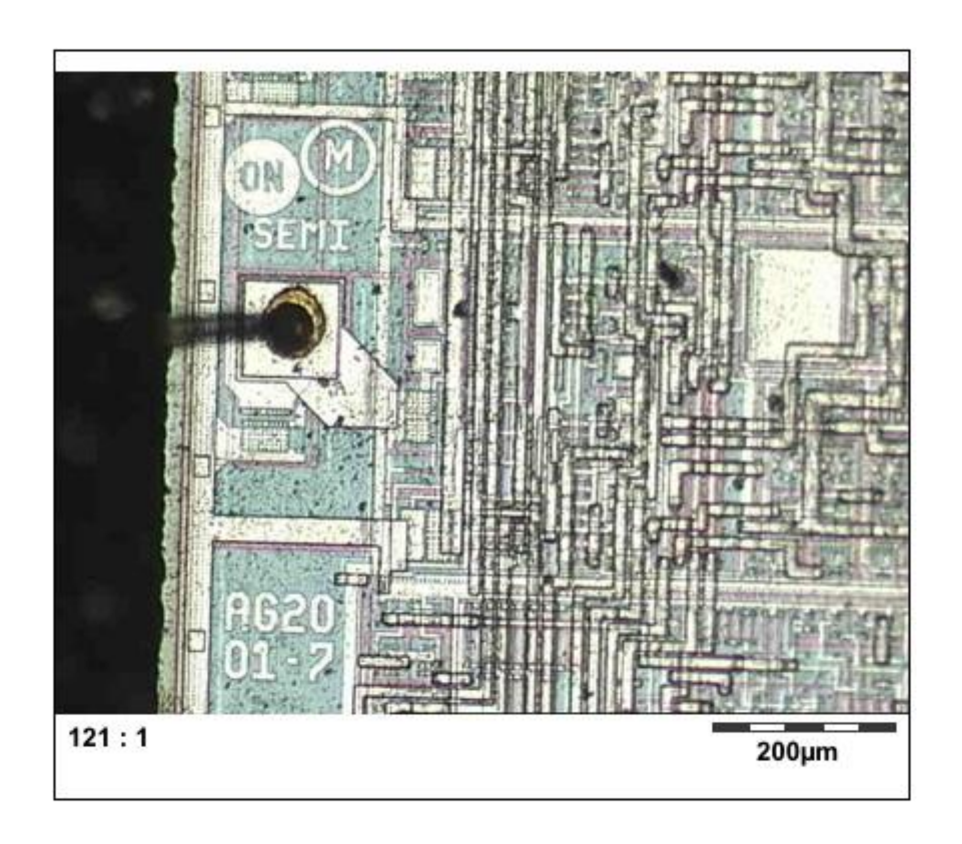


المكونات الأساسية داخل الدوائر المتكاملة هي طبقات معقدة من أشباه الموصلات (semiconductors)والنحاس وبعض المواد الأخرى تتصل معاً لتكوين ترانزستورات أو مقاومات أو أي مكونات أخرى في الدائرة. ويطلق على مجموعة تلك الطبقات المُشكّلة معاً اسم قالب(die)

نظرة عامة على قالب دائرة متكاملة :



الدوائر المتكاملة تكون ذات حجم ضئيل، لكن شرائح شبه الموصل وطبقات النحاس التي تُكون الدائرة المتكاملة تكون أصغر بكثير جداً، والتوصيلات بين تلك الطبقات معقدة للغاية. في الصورة التالية نرى قطاع مُكبر من القالب الموجود بالأعلى:

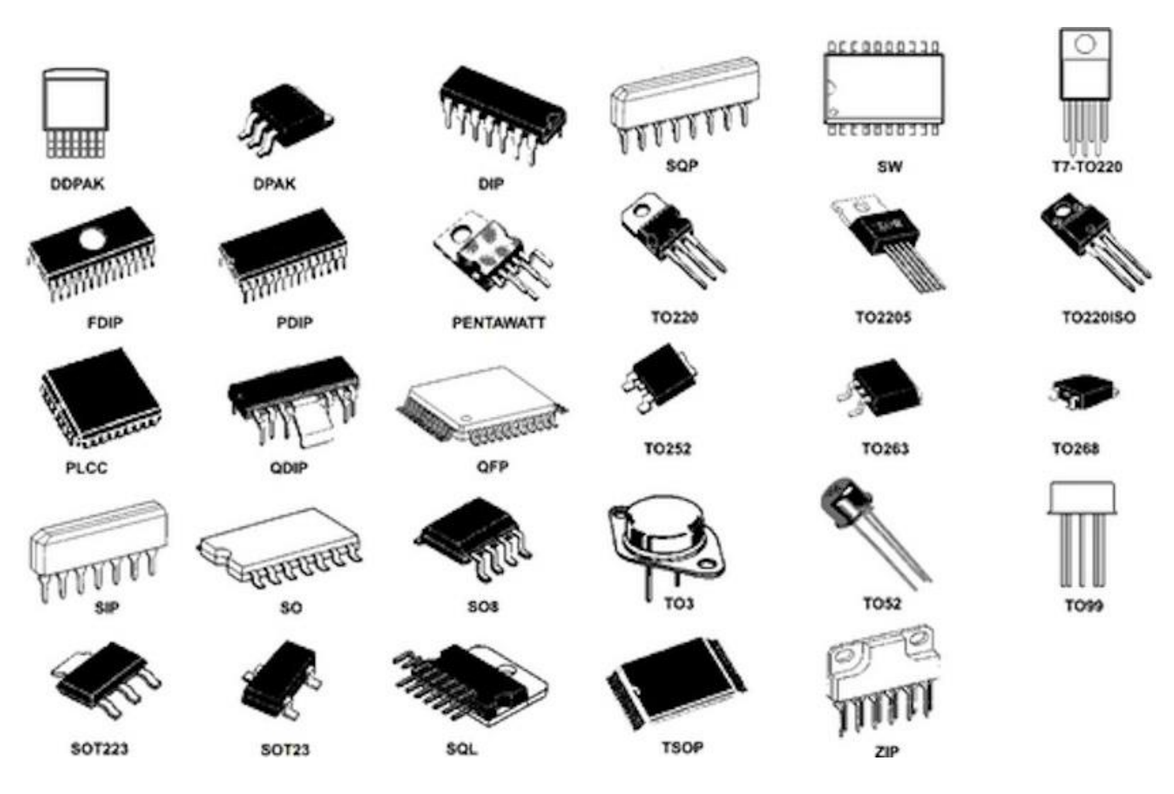



قالب الدائرة المتكاملة هو أصغر شكل لها، وهو صغير للغاية لدرجة لا تسمح بلحامه أو توصيله بمكونات أخرى. لكي نجعل توصيل أو لحام الدوائر المتكاملة أمراً سهلاً يتم تغليف القالب. حزم الدوائر المتكاملةIC packages اي الغطاء الخارجي, تحول القالب الحساس الضئيل إلى الرقاقة السوداء التي نراها في كل مكان.

حزم الدوائر المتكاملة (IC packages)

الحزمة هي الغلاف الذي يحتوي قالب الدائرة المتكاملة ويحوله إلى مكون يسهل توصيله بالأجهزة والمكونات الأخرى. يتم توصيل كل وصلة من الوصلات الخارجية للقالب بطرف (منفذ- سن (pin) (في الحزمة من خلال سلك دقيق من النحاس او الذهب. المنافذ هي الأطراف الفضية البارزة من حزمة الدائرة المتكاملة، والتي تعمل على توصيل الدائرة المتكاملة بباقي المكونات والأسلاك في الدوائر الإلكترونية. هذه المنافذ تمثل أهمية قصوى لنا عند التعامل مع الدوائر المتكاملة لأنه من خلالها نستطيع توصيل الدائرة المتكاملة بباقي المكونات والأسلاك في الدوائر الإلكترونية.

هناك العديد من أنواع حزم الدوائر المتكاملة، تختلف فيما بينها في الأبعاد وطريقة التثبيت وعدد المنافذ والخصائص الداخلية :

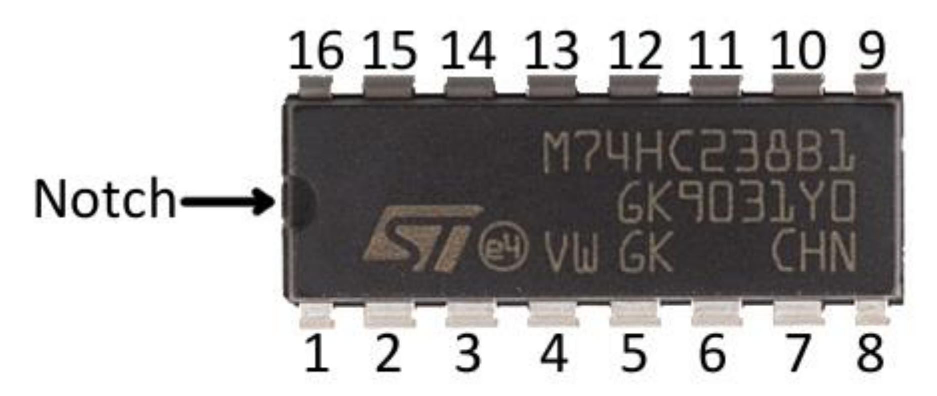


كيفية معرفة القطبية (polarity) وترقيم المنافذ

جميع الدوائر المتكاملة مكونات مستقطبة(polarized) ، وكل سن من السنون التي تبرز من الدائرة المتكاملة له وظيفة محددة وموضع معين. وهذا يعني أنه لا بد من وجود طريقة للتفريق بين تلك السنون وترقيمها. في معظم الدوائر المتكاملة يتم استخدام شق (notch) أو نقطة (dot) للإشارة إلى السن رقم 1. (أحيان يُستخدم كلاهما، وأحياناً يتم استخدام أحدهما فقط)




بعد أن تقوم بتحديد السن رقم 1 يتم ترقيم باقي السنون بحيث يزداد رقمها تدريجياً مع الحركة عكس عقارب الساعة حول الحزمة. الصورة التالية توضح ذلك بالتفصيل:



## طريقة التثبيت(mounting style)

أحد الخصائص المميزة لحزم الدوائر المتكاملة هي طريقة تثبيتها على ألواح الدوائر. جميع الحزم يتم تثبيتها بإحدى طريقتين: التثبيت عبر الثقوب through-hole) ((PTH)أو التثبيت السطحي .((surface-mount (SMD) الحزم التي يتم تثبيتها عبر الثقوب عادة ما تكون أكبر في الحجم وأسهل في الاستخدام. وهي مصممة لكي يتم تثبيتها على أحد جانبي اللوح (عن طريق إنفاذ سنونها خلال ثقوب معدة لذلك) ولحامها من الجانب الآخر.

الحزم السطحية يتراوح حجمها من الحزم الكبيرة إلى الحزم متناهية الصغر. وجميعها مصممة لكي يتم وضعها على أحد جانبي لوح الدائرة ولحامها على سطحه. السنون الخاصة بالحزم السطحية إما أن تكون بارزة من الجوانب عموديةً على الحزمة، وأحياناً تكون مرتبة على شكل مصفوفة أسفل الحزمة. الدوائر المتكاملة السطحية صعبة الاستخدام والتركيب يدوياً، وغالباً نحتاج لأدوات خاصة للتعامل معها وتثبيتها.

## حزمDIP

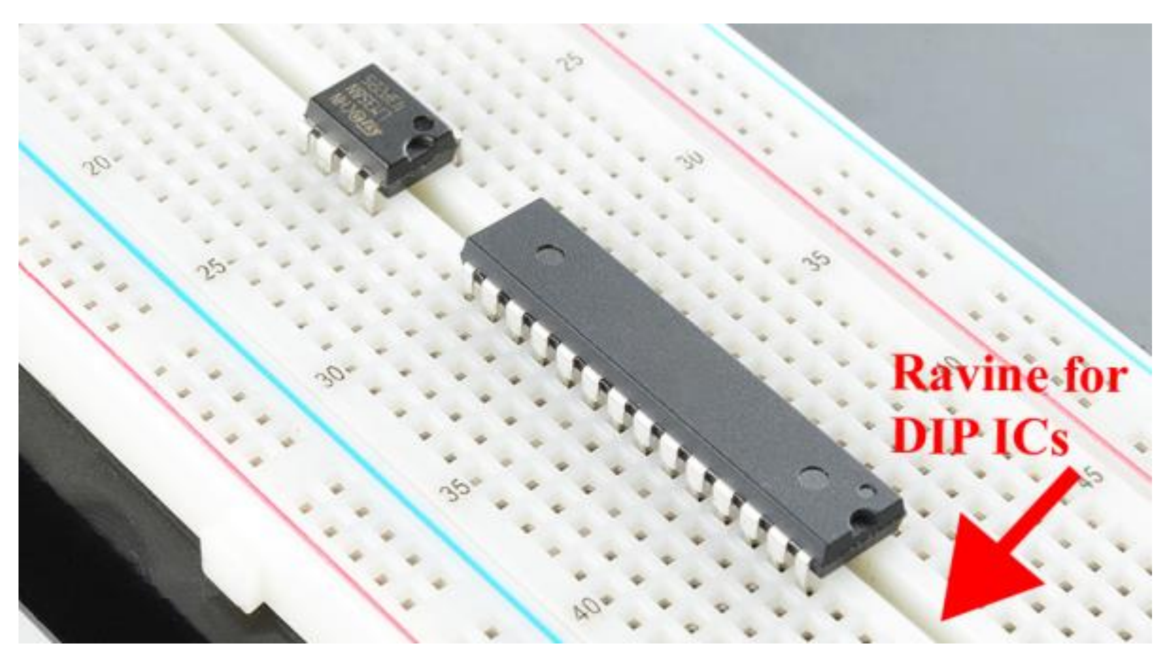
حزم DIP هي أكثر حزم الدوائر المتكاملة التي يتم تثبيتها عبر الثقوب شيوعاً واستخداماً. هذه الشرائح الصغيرة تحتوي على صفين متوازيين من السنون تبرز بشكل عمودي من غلاف مستطيل أسود بلاستيكي , مثال على ذلك متحكم دقيق ATmega328 ذو 28 سن هو أحد حزم DIP شائعة الاستخدام.



يفصل بين كل سنّين من سنون حزم الدوائر المتكاملة DIP 0.1 بوصة (2.54 ميليمتر) وهذه مسافة قياسية مناسبة لكي يتم تركيبها على ألواح التجارب

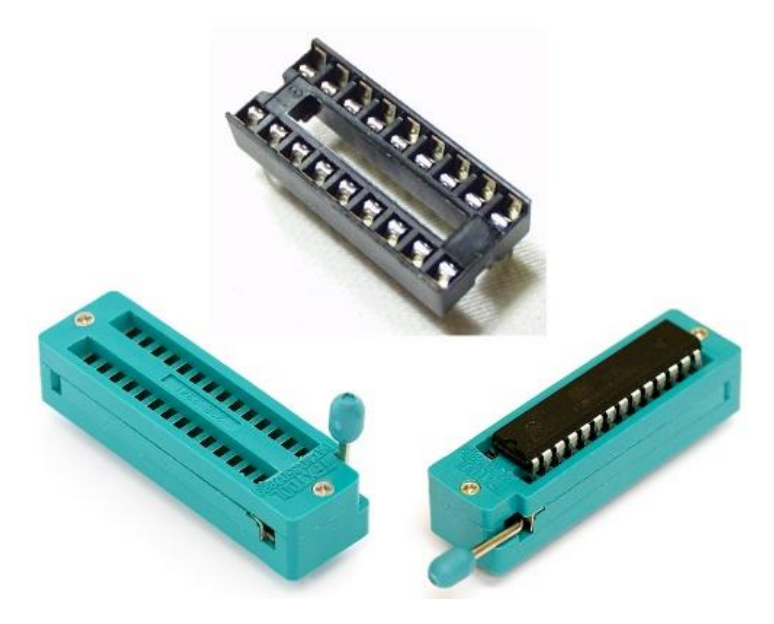
(breadboards)وألواح عمل النماذج الأولية الأخرى. تعتمد الأبعاد الكلية لحزم DIPعلى عدد السنون التي تحتويها، والتي يمكن أن تتراوح من أربعة إلى 64 سن.

المسافة الفاصلة بين صفي السنون في حزم DIP مُعينة بدقة لكي تتناسب مع ألواح التجارب حيث يتم تثبيتها حول الأخدود الموجود في المنطقة المركزية من لوح التجارب. وهذا يعمل على الفصل بين سنون كل صف وعدم حدوث قصر.(short)

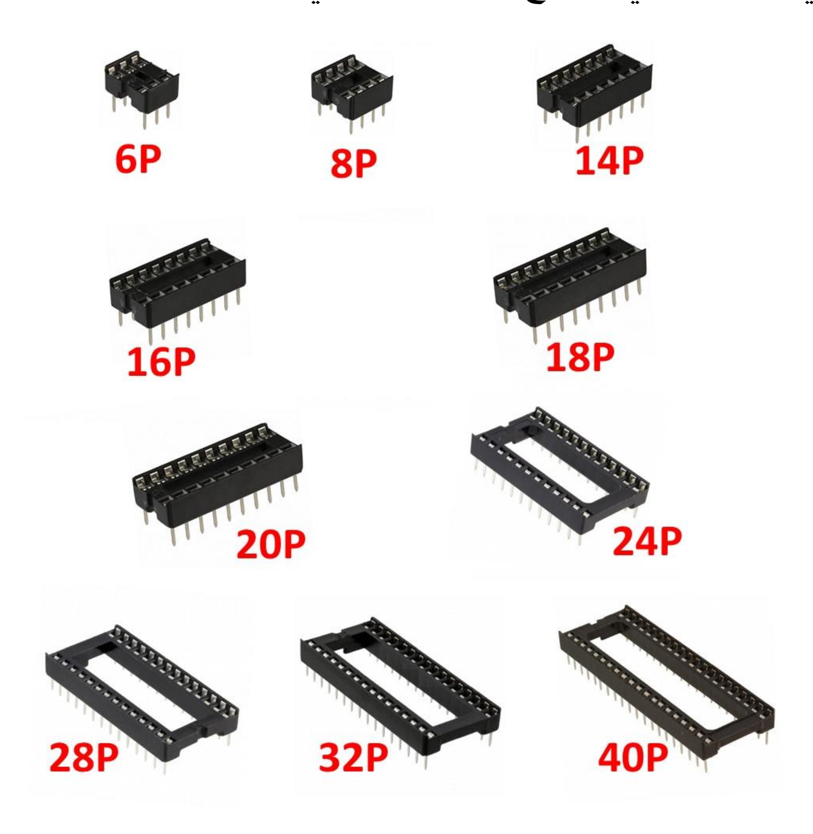



بالإضافة إلى استخدامها مع ألواح التجارب، يمكن أيضاً لحام دوائر DIP المتكاملة في ألواح الدارات المطبوعة (PCBs)، حيث يتم إدخال سنون الحزمة من أحد جانبي اللوح ويتم لحامها على الجانب الآخر. في بعض الأحيان بدلاً من لحام الدائرة المتكاملة مباشرة في اللوح يتم عمل مقبس (socket) لها يمكن من خلاله إزاله حزم DIP أو استبدالها إذا كانت هناك حاجة لذلك. (أي يتم لحام المقبس بدلاً من الحزم ويتم توصيل الحزمة بالمقبس بطريقة تسمح بإزالتها.)

مقبس عادي لحزم DIP ومقبس ZIF مع وبدون دائرة متكاملة.



وفي الشكل التالي نوضح عدد المنافذ التي يمكن توصيلها بالمقبس



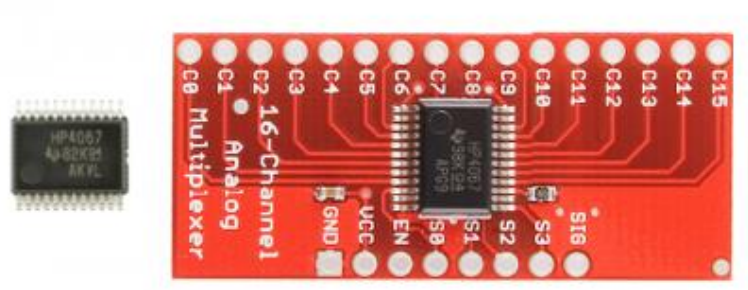
# الحزم السطحية(SMD Packages)

يوجد حالياً العديد من أنواع الحزم السطحية. لاستخدام حزمة سطحية من الدوائر المتكاملة تكون هناك حاجة لاستخدام لوحة دارات مطبوعة مصنوعة خصيصاً لتلك الحزمة، أي تحتوي على قالب نحاسي مناسب للحزمة حتى يتم لحامها وتثبيتها عليه.

والآن سنتعرف على بعض أنواع الحزم السطحية شائعة الاستخدام، والتي تتدرج من ناحية سهولة لحامها من "سهلة" إلى تلك التي "تتطلب بعض الأدوات الخاصة" إلى التي "تتطلب أدوات خاصة جداً تعمل عادة بشكل آلي".

## حزمSOIC

حزم SOIC السطحية هي المماثل السطحي لحزم DIP. فهي ما ستحصل عليه إذا قمت بثني جميع سنون حزمة DIP للخارج مع تصغير حجمها. هذه الحزم تُعتبر من أسهل أنواع الحزم السطحية في اللحام والتثبيت؛ فهي لا تحتاج سوى ليد ثابتة والنظر بتركيز عن قرب. يفصل بين كل سنين متتاليين من السنون الموجودة في حزم SOIC مسافة قدرها 0.05 بوصة (mm1.27).





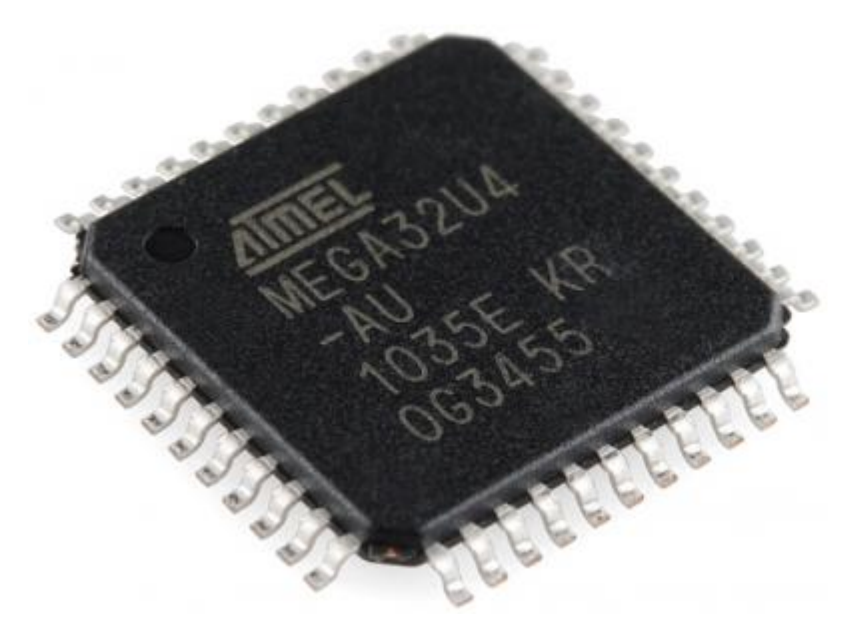


مضاعف إرسال (multiplexer) ذو 16 قناة (CD74HC4067) على شكل حزمة SSOP ذات24سن مثبتة في منتصف دائرة لوحة دوائر مطبوعة. (مقارنة مع حجم عملة معدنية)

الحزم الرباعية المسطحة (Quad Flat Packages (QFPs))

فلطحة سنون الدائرة المتكاملة في الأربعة اتجاهات ينتج عنه الحزم الرباعية المسطحة (QFPs). حزم الدوائر المتكاملة QFP يمكن أن تحتوي على سنون تتراوح من ثمانية لكل جانب (مجموعها 32) إلى ما يزيد عن سبعين سن لكل جانب (مجموعها قد يتجاوز 300). المساحة الفاصلة بين السنون في حزم QFP عادة ما تتراوح بين 0.4 mm إلى 1 mm. هناك أشكال أخرى من حزم QFP القياسية ولكنها أصغر في الحجم مثل الحزم النحيفة (TQFP) والحزم شديدة النحافة (VTQFP) وحزم LQFP.

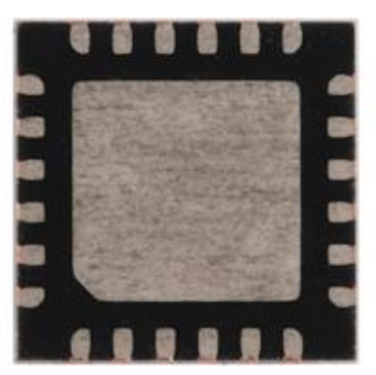
المعالج الدقيق ATmega32U4 في حزمة TQFP تحتوي على 44 سن (11 في كل جانب) :




إذا قمت بصقل السنون الخاصة بدائرة متكاملة QFP فستحصل على حزمة رباعية مسطحة بدون سنون (quad-flat no-leads (QFN)). الوصلات الموجودة بحزمة QFN هي عبارة عن مساند (pads) صغيرة مكشوفة على الحواف الجانبية السفلية من الدائرة المتكاملة، وأحياناً تكون على الجوانب وفي الأسفل، وفي حزم أخرى تكون المساند في أسفل الحزمة فقط.

مستشعر IMU MPU-6050 متعدد الاستخدامات يأتي على شكل حزمة QFN دقيقة تحتوي على 24 سن مصقول موجود على الحواف السفلية من الدائرة المتكاملة:



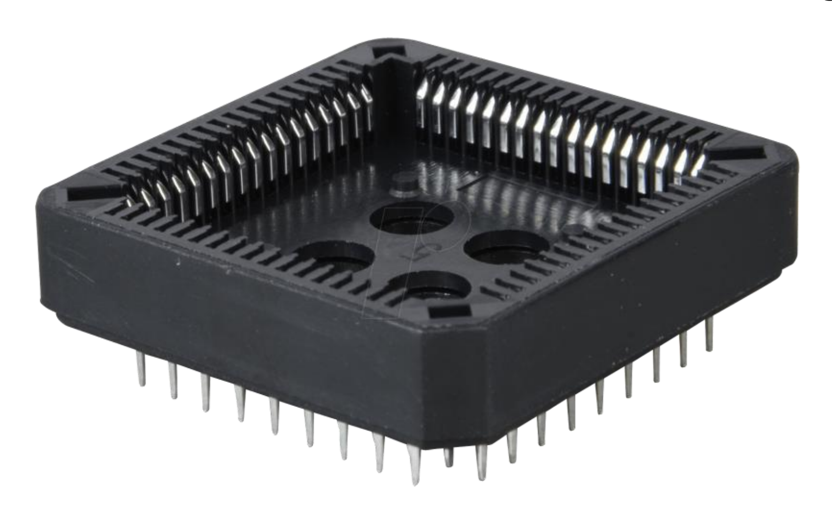






هناك أشكال أخرى مصغرة من حزم QFN القياسية مثل الحزم النحيفة(TQFN) ، والحزم شديدة النحافة (VQFN) وحزم .(MLF) وهناك أيضاً حزم ألواح متكاملة تحتوي على سنون على جانبين من جوانبها فقط مثل حزم DFN وحزم.TDFN

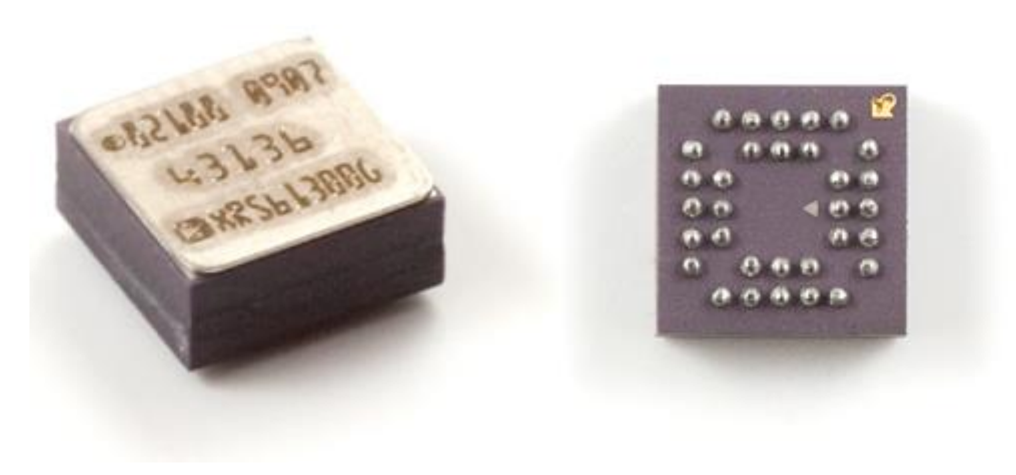
الكثير من المعالجات الدقيقة والمستشعرات والدوائر المتكاملة الحديثة تأتي على شكل حزم QFP أو QFN. على سبيل المثال المعالج الدقيق ATmega328 شائع الاستخدام يأتي على شكل حزمة TQFP أو MLF، بينما مقياس التسارع/جيرسكوب مثل MPU-6050 يأتي في حزمة QFN صغيرة الحجم.

شكل المقبس :



## شبكات الكرات المصفوفة(Ball Grid Arrays)

حزم شبكات الكرات المصفوفة (BGA) هي عبارة دوائر متكاملة متقدمة ومتطورة للغاية. فهي عبارة عن حزم صغيرة شديدة التعقيد تبرز من أسفلها كرات صغيرة من معدن لحامي مرتبة في شبكة ثنائية الأبعاد، وأحياناً تكون تلك الكرات متصلة مباشرة مع قالب الدائرة المتكاملة.



تعتبر حزم BGA مخصصة في الغالب للمعالجات الدقيقة المتطورة مثل تلك الموجودة في بطاقات بي سي دوينو (pcDuino) أو بطاقات راسبيري باي Raspberry Pi

إذا كان بإمكانك لحام حزم الدوائر المتكاملة BGA يدوياً فأنت إذن لديك مهارة فريدة. في الغالب يتطلب تثبيت تلك الحزم على لوحات الدوائر المطبوعة PCBs استخدام آلات خاصة مثل آلات تثبيت المكونات السطحية (pick-and-place machines) وأفران التدفق.(reflow ovens)

لمعرفة انواع الدوائر المكاملة انظر الملحق A0

IC Package Types

تنتشر الدوائر المتكاملة بأشكالها المختلفة في العديد من الإلكترونيات التي يصعب حصرها. ولكننا سنكتفي بذكر بعض أكثر أنواعها شيوعاً والتي ستقابلها أثناء دراستك للإلكترونيات في هذه السلسلة :

1 - المستشعرات (sensors)

المستشعرات الرقمية الحديثة مثل مستشعرات الحرارة ومقاييس التسارع والجيرسكوب غالبها تأتي على شكل حزم دوائر متكاملة.

هذه الدوائر المتكاملة عادة ما تكون أصغر من المتحكمات الدقيقة أو الدوائر المتكاملة الأخرى المثبتة على ألواح الدوائر المطبوعة، ويتراوح عد السنون بها من 3 إلى 20 سن. أصبح من النادر أن نرى مستشعر على شكل حزمةDIP ، والغالبية العظمى حالياً حزم QFP أو QFN أو حتى.BGA

2 - المتحكمات الدقيقة (microcontrollers)، المعالجات الدقيقة (microprocessors)

المتحكمات الدقيقة والمعالجات الدقيقة ومصفوفات البوابات المنطقية القابلة للبرمجة جميعها تحتوي على ملايين أو مليارات الترانزستورات في رقاقة إلكترونية دقيقة بداخل الدوائر المدمجة. هذه المكونات لها مدى كبير من الوظائف والتعقيد والأحجام؛ تتدرج في مواصفاتها من المتحكم الدقيق بسعة 8 بت مثل ATmega328 الموجود في بطاقات أردوينو، إلى المعالجات الدقيقة متعددة الأنوية 64 بت المعقدة التي تستخدم في أجهزة الحاسب الآلي.

هذه المكونات عادة ما تكون هي أكبر دائرة متكاملة في الدائرة الإلكترونية. المتحكمات الدقيقة البسيطة تأتي في حزم DIP أو QFN أوQFP ، ويتراوح عدد السنون في الحزمة بين 8 و 100. وكلما زاد تعقيد هذه المكونات زاد تعقيد الحزم أيضاً. مصفوفات البوابات المنطقية القابلة للبرمجة والمعالجات الدقيقة المعقدة يمكن أن تحتوي على ما يزيد عن ألف سن وتأتي على شكل حزم متطورة فقط مثل QFN أو LGA أو.BGA

اذا تبقى لنا الان هو ان نبدأ في التعمق باستخدام الدوائر المتكاملة هيا بنا نبدأ رحلة التعلم سندرس في هذا الكتاب العديد من الدوائر المتكاملة لمعرفة خصائصها وكيف نعمل بها :

المؤقتات(Timers)

هو تماما كما تنص عليه الكلمة "مؤقت" اي ببساطة هو ان تنتظر waits فترة زمنية محددة قبل فعل شيء ما , وهو نوع خاص من الساعات الزمنية ، تستخدم للتحكم في تسلسل العمليات الإنتاجية ، كما تستخدم ساعات الإيقاف مثلا لقياس الوقت اللازم لإنهاء عملية ما ابتداءاً من الصفر إلى انتهاء زمن العملية .

أنواع المؤقت الزمني Timer :

1 -المؤقت الزمني الميكانيكي.

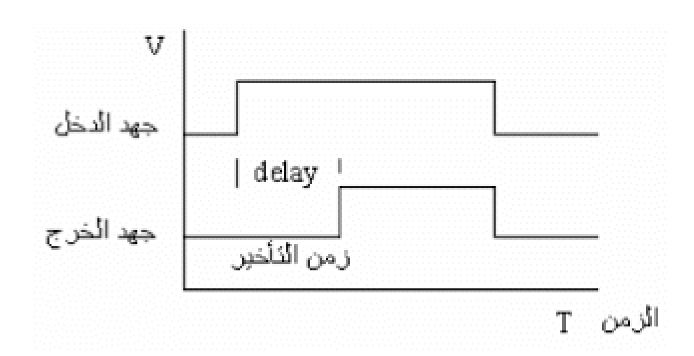
2 -المؤقت الزمني الكهروميكانيكي.

3- المؤقت الزمني الإلكتروني .

أنواع المؤقتات من حيث نوع الخرج:

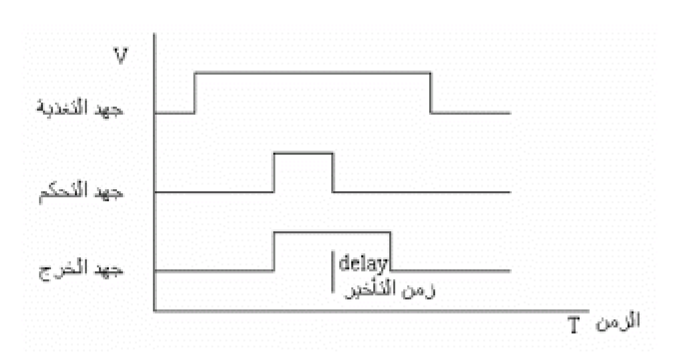
1 -المؤقت الزمني الذي يعطي خرج بعد مرور زمن معين ويسمى : On Delay Timer

عندما يوصل هذا المؤقت بالتيار المقنن فإنه لا يعطي خرجاً إلا بعد مرور فترة زمنية محددة ( تم ضبط المؤقت عليها ) ، ثم يستمر هذا الخرج إلى أن يتم فصل التيار عن المؤقت فينقطع الخرج في نفس اللحظة. ويستعمل هذا المؤقت On deley timer في دوائر التحكم في بدء حركة محرك.



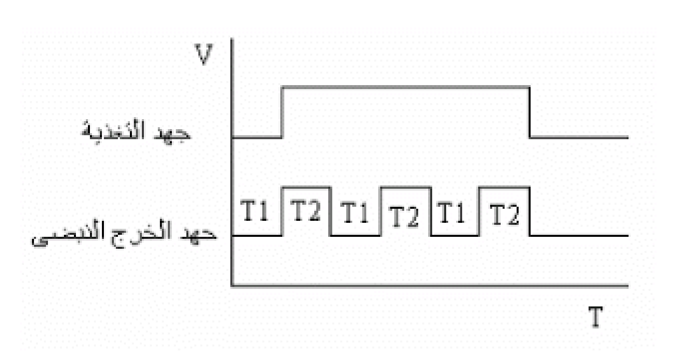

2 - المؤقت الزمني الذي يستمر خرجه حتى بعد فصل مفتاح التحكم عنه ويسمى : OFF Delay

ويجب تغذية هذا المؤقت دائماً بالتيار الكهربي ، ويتم تشغيله عن طريق دخل إضافي ( جهد التحكم ) ، فعند توصيل جهد التحكم بالمؤقت فإنه يعطي خرجاً في نفس اللحظة ، وعند فصل جهد التحكم عنه يستمر هذا الخرج لفترة زمنية محددة ( تم ضبط المؤقت عليها ) ، ثم بعد ذلك ينقطع الخرج. ويستعمل هذا المؤقت Off deley timer في دوائر التحكم في إنارة سلم عمارة

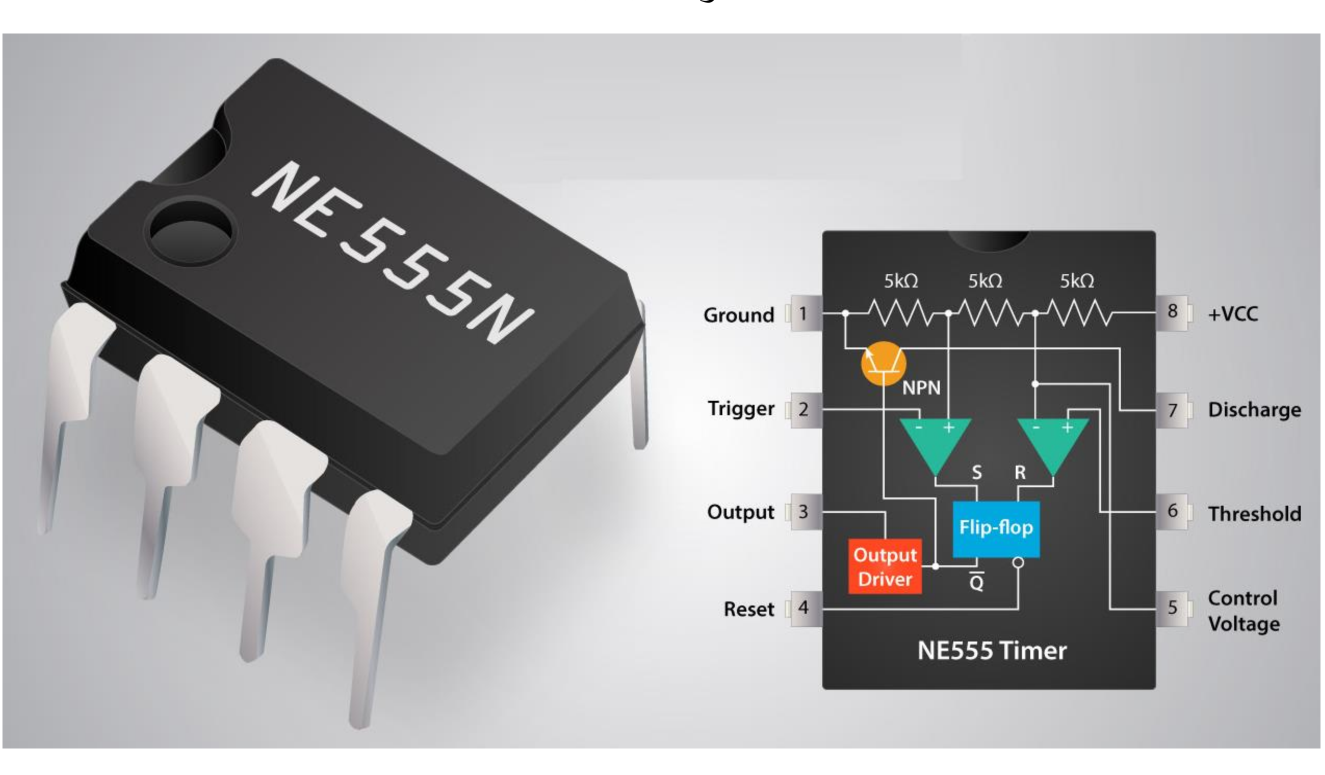


3 - المؤقت النبضي ويسمىPulse Timer :

ويجب أن يغذى أيضاً المؤقت بالتيار الكهربي طول فترة التشغيل ، وخرج هذا المؤقت عبارة عن سلسلة من النبضات طول زمن فترة كل منها يمكن ضبطه .



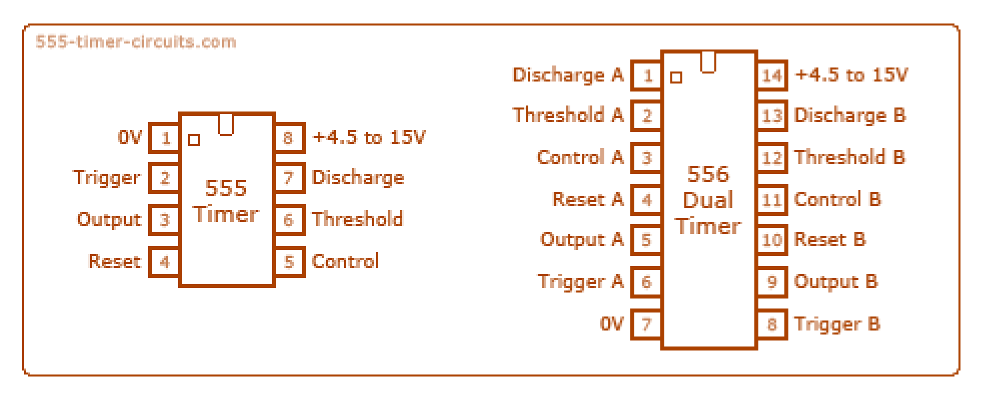
# المؤقت 555




المؤقت 555 عبارة عن دائرة متكاملة بسيطة يمكن استخدامها لإجراء العديد من الدوائر الإلكترونية المختلفة. مع هذه المعلومات سوف تتعلم كيف تعمل 555 وسوف يكون لديها الخبرة لبناء بعض الدوائر.

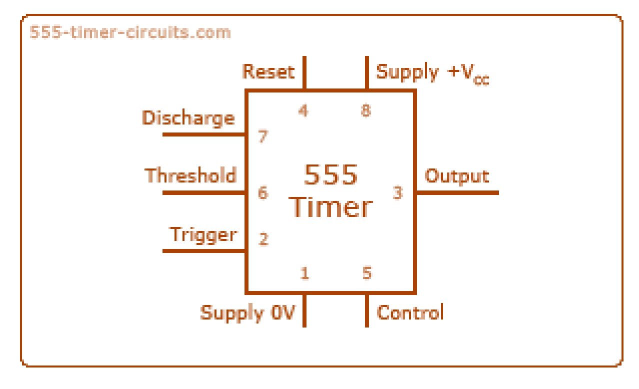
تعتبر الدائرة المتكاملة 555 (IC) جهاز توقيت سهل الاستخدام يحتوي على العديد من التطبيقات. يستخدم على نطاق واسع في الدوائر الإلكترونية . كما يتوفر إصدار "مزدوج" يُسمى 556 والذي يتضمن جهازي 555 مستقلاً في حزمة واحدة.

يوضح الرسم التوضيحي التالي كلا من 555 (8 سن) و 556 (14 سن).




ولكي تعمل 555 فإنها تعتمد على التقنيات الإلكترونية التناظرية والرقمية ، ولكن إذا أخذنا في الاعتبار ناتجها فقط ، فيمكن اعتبارها كجهاز رقمي. يمكن أن يكون الإخراج في إحدى الحالات ما يلي ، الحالة الأولى هي حالة "منخفضة" ، وهي v0. الحالة الثانية هي الحالة "عالية" ، وهي الجهد الكهربي (جهد التيار الكهربائي الخاص بك والذي يمكن أن يكون أي شيء من 4.5 إلى 15 فولت. 18 فولت كحد أقصى).

نريد ان نرسم المخطط بطريقة اخرى حتى يسهل علنا دراسة الدائرة




لاحظ كيف أن الدبابيس ليست بنفس ترتيب الشريحة الفعلية ، وذلك لأنه من السهل التعرف على وظيفة كل دبوس كما سترى لاحقا

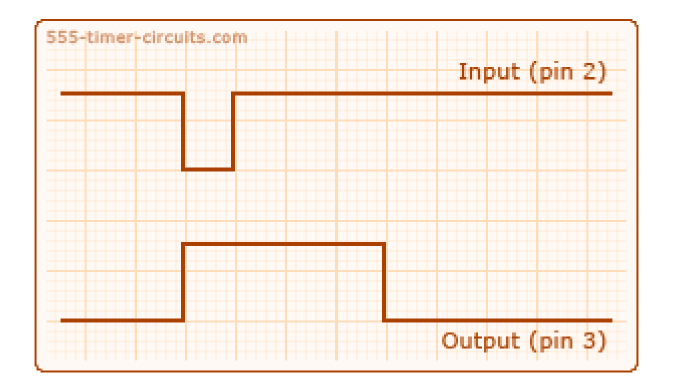
يمكن تصنيف أنواع المخرجات او ما يسمى بالحالات الثلاثة الأكثر شيوعًا اي اوضاع التشغيل من خلال ما يلي (أسماءهم وفكرة عن وظائفهم):

1 –الوضع وحيد الاستقرار Monostable:

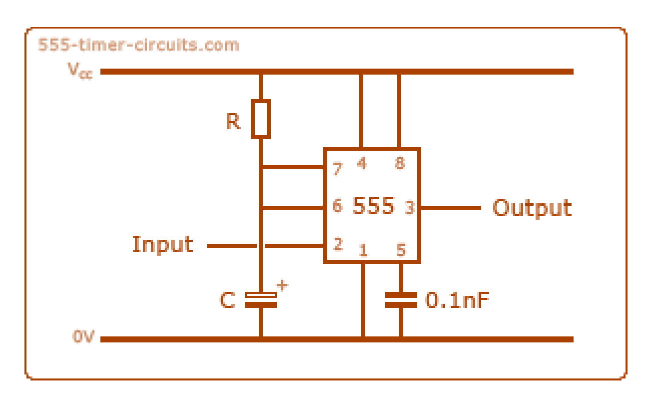
في الوضع وحيد الاستقرار لدينا مدخل واحد عند تغيير حالته من 1 الى 0 تتغير حالة المخرج من 0 الى 1 و يبقى في الحالة 1 لمدة معينة سوف نرى في الفقرة التالية كيف نحدد هذه المدة باستعمال المكونات الالكترونية المحيطة بالمؤقت 555.

و باختصار يمكن استعمال هذه الطريقة في مشروع فيه مدخل واحد كالزر مثلا حين يضغط المستعمل على هذا الزر يعمل المخرج لمدة معينة ثم يتوقف تلقائيا.

شاهد الصورة التالية لفهم كيفية عمل الوضع وحيد الاستقرار.في الاعلى نلاحظ حالة المدخل و في الاسفل حالة المخرج:




الصورة التالية تبين كيفية ربط المؤقت بمكثف C و مقاوم R من اجل الحصول على الوضع وحيد الاستقرار.




من أجل حساب المدة التي يبقى فيها المخرج مفعلا يمكن استعمال المعادلة التالية

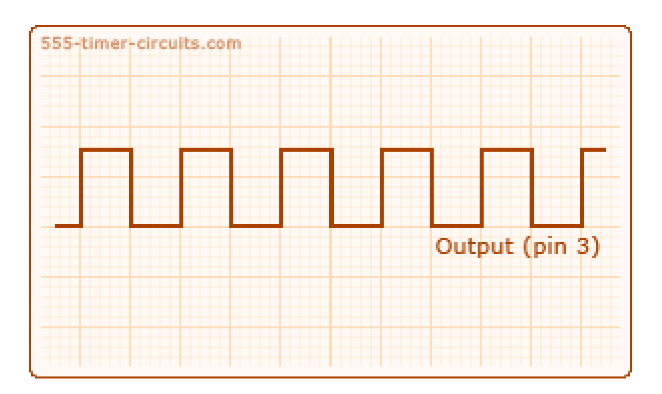
Time Out Delay (secs) = 1.1 * R1 * C1

ملاحظة : يمكن استعال مقاوم متغير من اجل تغيير المدة اثناء عمل المشروع, نفس الشيء كذلك بالنسبة للوضع غير المستقر.

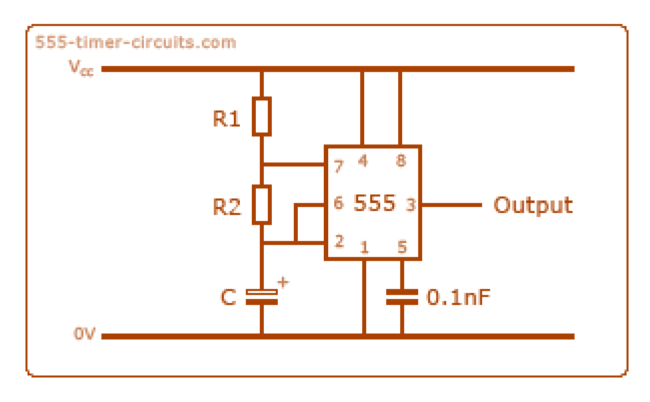
وظائف 555 هنا بمثابة "طلقة واحدة" one-shot. وتشمل التطبيقات أجهزة ضبط الوقت ، وفقدان اكتشاف النبض ، ومفاتيح bouncefree ، والمفاتيح اللمسية ، ومقسم التردد ، وقياس السعة ، وتضمين عرض النبضة (PWM) ، الخ

2 - الوضع الغير مستقر Astable - free running:

كما يوضح اسم هذه الطريقة فإن المخرج في هذه الحالة يكون مرة في حالة 0 و مرة في حالة 1 بطريقة متتالية او ما يسمى بالموجة المربعة (square ‘wave‘) كما توضح الصورة التالية:




ومن أجل الحصول على مخرج بهذا الشكل اي على شكل موجة مربعة يجب اضافة مقاومان R1 و R2 و مكثف C و ربطها مع المؤقت 555 بهذا الشكل .



من أجل حساب قيمة المقاومات و المكثف للحصول على خاصيات محددة للموجة المربعة يمكن استعمال المعادلات التالية:

حساب تردد الموجة المربعة(frequency)

(Frequency = 1.44 / ((R1 + R2 + R2) * C1

حساب وقت الحالة 1 ووقت الحالة 0 في التردد الواحد

Time High (secs) = 0.693 * (R1 + R2) * C1

Time Low (secs) = 0.693 * R2 * C1

النسبة المئوية لدورة العمل

Duty Cycle Percentage = (Th / (Th + Tl)) * 100

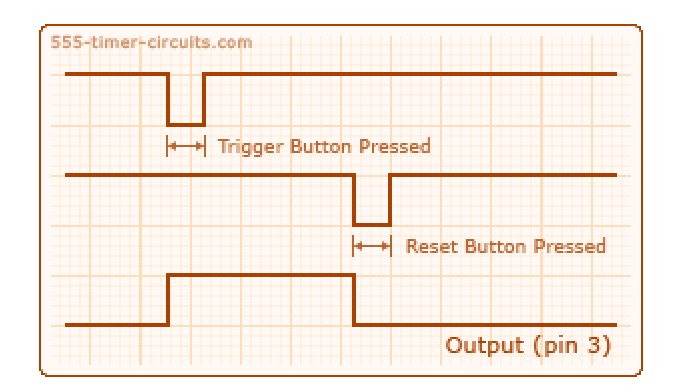
يمكن استعمال هذا الوضع للمؤقت 555 في العديد من التطبيقات مثل اشعال و اطفاء مصباح LED يمكن استعماله كذلك من اجل التحكم في سرعة محرك التيار المستمر. كما يمكن ان يكون عبارة عن ساعة clock أو ذبذبة لشرائح الكترونية أخرى كالعداد

يمكن تشغيله كمؤشر مذبذب. وتوليد النبضات ، الساعات المنطقية ، توليد النغمة ، اجهزة الإنذار الأمنية ، تعديل وضع النبضة PWM ، إلخ.

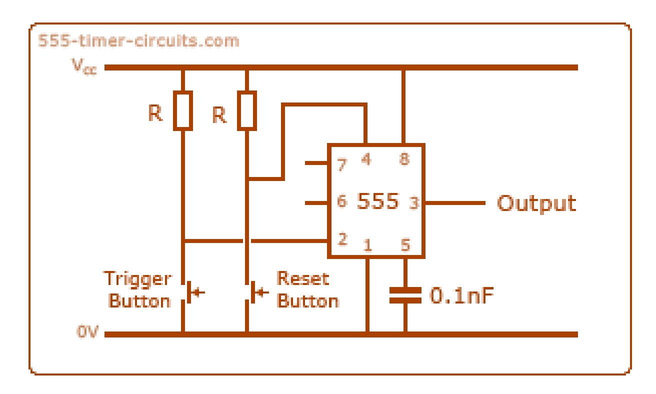
3 - الوضع ثنائي الاستقرار Bistable mode trigger

الوضع ثنائي الاستقرار شبيه بالوضع وحيد الاستقرار, الا ان الفرق بينهما هو ان المخرج في الاحادي الاستقرار ينتهي من التفعيل بعد انتهاء مدة محددة كما شاهدنا, اما في الوضع ثنائي الاستقرار فان المخرج يعود من الحالة 1 الى 0 بعد بعد تفعيل مدخل ثاني غير المدخل الاول المتسبب في بدا التفعيل. نأخذ مثالا بسيطا كي يتضح الحال , نأخذ محرك كمخرج و ناخذ زرين كمدخلين . عندما يضغط المستعمل على الزر الاول يشتغل المحرك و لا يتوقف المحرك الا اذا ضغطنا على الزر الثاني.

للفهم اكثر شاهد صورة الاشارات التالية للمدخلين و المخرج:

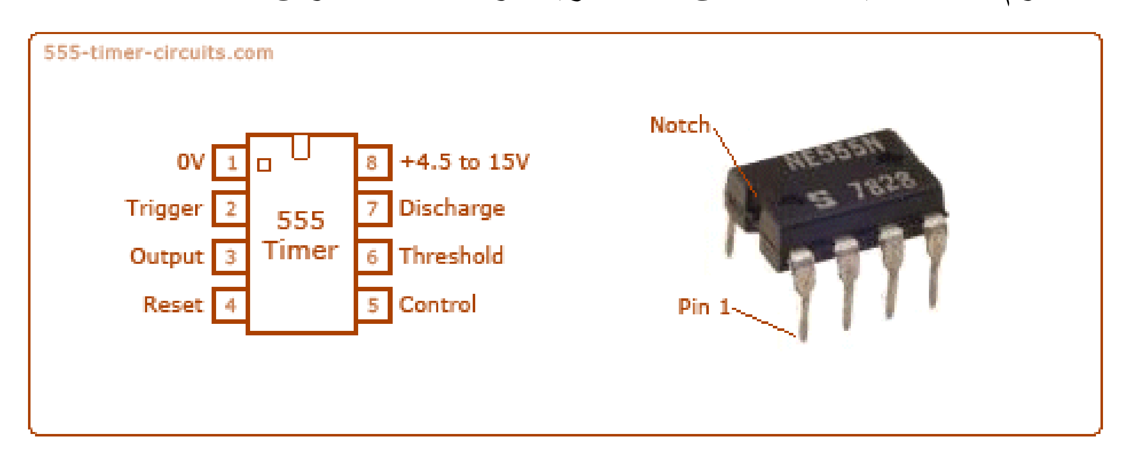



و الصورة التالية توضح كيفية ربط المؤقت 555 من اجل استعماله في الوضع ثنائي الاستقرار.



وظائف دبابيس المؤقت 555

هنا سنقوم بعمل تحليل القطعة من اجل تعريف وظائف كل دبوس:




الطرف 1: الأرضي GND ground

الطرف 2 : البدء (القدح – الإشعال ) Trigger, سمِي بهذا لأنه يعمل مثل مسدس بداية و الذي يجعل (555) تبدأ بالعمل. و هذا الزناد لا يعمل إلا عند إنخفاض فرق الجهد (الفولت) بمقدار ثلث القوة اللتي تعمل بها مثال ثلث ال(9 فولت) هو (9*3/1) و اللذي يساوي (3) ففي حالة أن فرق الجهد (الفولت) قل عن الثلاثه فيبدأ الزناد بالعمل و يجعل المخرج رقم (3) في حاله الموجب (+) .

إذا كان الدبوس 2 LOW، ودبوس 6 LOW ، فسيكون المخرج HIGH

إذا كان دبوس 6 HIGH ، والدبوس 2 LOW، يخرج الإخراج LOW

هذا دبوس له مقاومة عالية جدا (حوالي M10) ، وسوف يقدح حوالي uA1.

الطرف 3 : يمثل المخرج Output ويكون هناك حالتان فقط لهذا المخرج و هما السالب (حوالي 0.5 فولت أقل من 0 فولت) أو الموجب الذي يقارب الرقم اللذي تعمل به مثلا بطاريه 9 فولت فستجد أقل من ال9 بقليل (حول V2 احيانا!) ولكن الخرج الحقيقي من

هذا المنفذ هو كم من الوقت هي موجبة و كم من الوقت هي سالبة واللتي تعتمد أساسيا علي المخارج الخمسة القادمة , ويخرج ما يصل إلى mA200.

الطرف 4 : إعادة التشغيل Reset والتي يمكن أن تستخدم لإعادة تشغيل توقيت رقائق (555) يجب أن تكون متصله بإمدادات التيار الكهربائي للعمل ، لأنه إذا أتصلت بالسالب وحدث أي خطأ في المؤقت لن يعمل مجددا حتي يأتيه إشارة من مخرج الزناد رقم (2) HIGH يجب أن تؤخذ v0.8 لإعادة الرقاقة.

الطرف 5 : التحكم Control , يتم توصيل هذا المخرج إلي السالب (v0) وعادة من خلال مكثفً صغير بقيمه (0.01) ميكرو فاراد (و الغرض من المكثف هو الحد من أي تقلبات في امدادات التيار الكهربائي التي قد تؤثر على عمل مؤقت فيجعله يعمل بسلاسه و بدون أي تغيرات في دورة الوقت للمؤقت) سيختلف الجهد المطبّق على هذا الدبوس من اعدادات توقيت دائرة RC (بشكل كبير).

الطرف 6 : يسمى احيانا باللسان Threshold و الغرض من هذا المخرج هوا التحكم في الفولت الذي يفرغ المكثف من قِبل مخرج التفريغ (7) يكون المخرج LOWفقط اذا كان 2 HIGH حيث أنه إذا وصل الجهد ثلثي امدادات التيار الكهربائي (VCC) ، تنتهي دورة التوقيت و يعود المخرج (3) إلي وضعه (0 فولت).

الطرف 7 : التفريغ Discharge وهذا المخرج يستخدم في تفريغ مكثف خارجي الذي يعمل جنبا إلى جنب مع المقاومة الخارجيه أيضا للسيطرة على الفاصل الزمني في التوقيت ، في معظم الدوائر، يتم توصيل هذا المخرج لامدادات التيار الكهربائي من خلال المقاومة الخارجية وإلى السالب من خلال مكثف خارجي أيضا.

يكون LOW إذا وصل الجهد الى الثلثين في طرف 6 لكن يجب ان يكون 2 HIGH

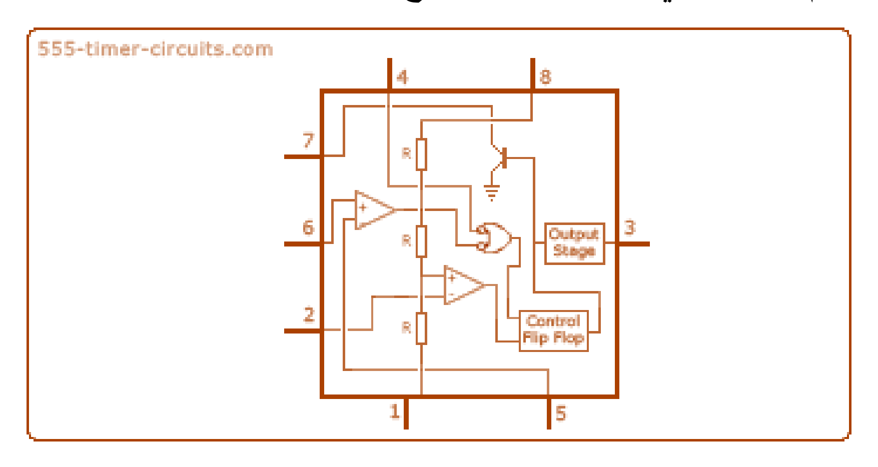
اذا كان الطرف2 HIGH , الطرف 6 يمكن ان يكون LOWاو HIGH لكن 7 يبقى LOW

يكون HIGH ويبقى كذلك عندما يكون الطرف 2 يعمل بمقدار ثلث القوة كنبضة منخفضة LOW pulse عندما يكون 6 LOW

الطرف 8 : التغذية DC Supply يتم توصيله إلى امدادات التيار الكهربائي الموجب ويجب أن يكون فرق الجهد على الأقل (4.5 فولت) ولا يزيد عن (15 فولت) ، من المتعارف عليه هو إستخدام أربعة بطاريات سعة AA أو AAA ، لتوفر (6 فولت) أو بطارية (9 فولت) واحدة ولكن عادة ما يكون V DC5 عند العمل مع الدوائر المتكاملة الرقمية.

المؤقت 555 من الداخل

قد تتسائل ما هو داخل رقاقة توقيت 555 أو ما يجعله يعمل. حسناً ، شريحة 555 مؤقِّتة IC وبالتالي فهي تحتوي على دائرة محاطة بالسيليكون. يتم توصيل كل من الدبابيس إلى الدائرة التي تتكون من أكثر من 20 ترانزستور ، 2 ديودات و 15 مقاومة , يوضح الرسم التوضيحي أدناه مخطط القطع الوظيفية للموقت 555 IC5:




هل تلاحظ وجود ثلاثة مقاومات والتي قيمتهم K5 ؟ هكذا حصلت الشريحة على اسمها.

استخدام مخرج مؤقت 555

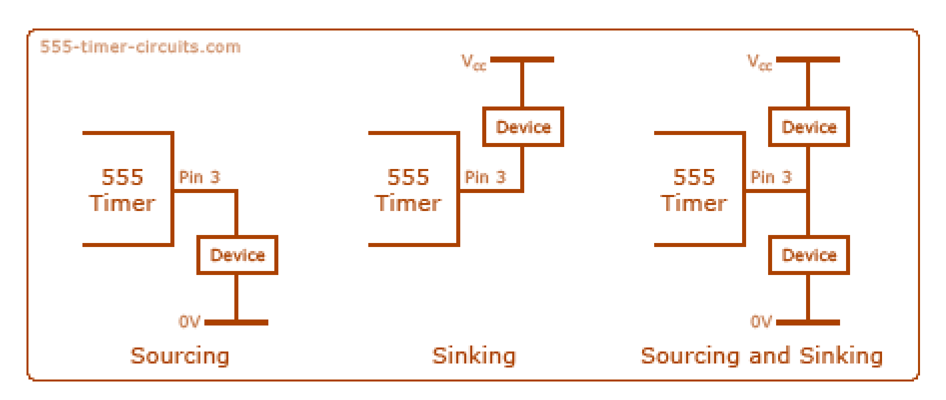
يمكن أن يكون المخرج (Pin 3) من 555 حالتين في أي وقت ، مما يعني أنه ناتج رقمي يمكن توصيله مباشرة بمدخلات ICs الرقمية الأخرى او الدوائر الالكترونية أو يمكنه التحكم في الأجهزة الأخرى بمساعدة بعض المكونات الإضافية, الحالة الأولى هي الحالة "المنخفضة" LOW، وهي V0 الجهد في التيار الكهربائي والحالة الثانية هي الحالة "عالية" HIGH ، وهي VCC الجهد في التيار الكهربائي.

## حالات ال Sinking and Sourcing

عندما يكون الخرج LOW ، سوف يتدفق التيار عبر الجهاز ويقوم بتشغيله وهذا ما يسمى "sinking" لأنه يتم الحصول على التيار من Vs ويتدفق عبر الجهاز و555 إلى V0.

عندما يكون الخرج HIGH سوف يتدفق التيار عبر الجهاز ويقوم بتشغيله وهذا ما يسمى "sourcing" لأنه يتم الحصول على التيار من 555 ويتدفق عبر الجهاز إلى V0.

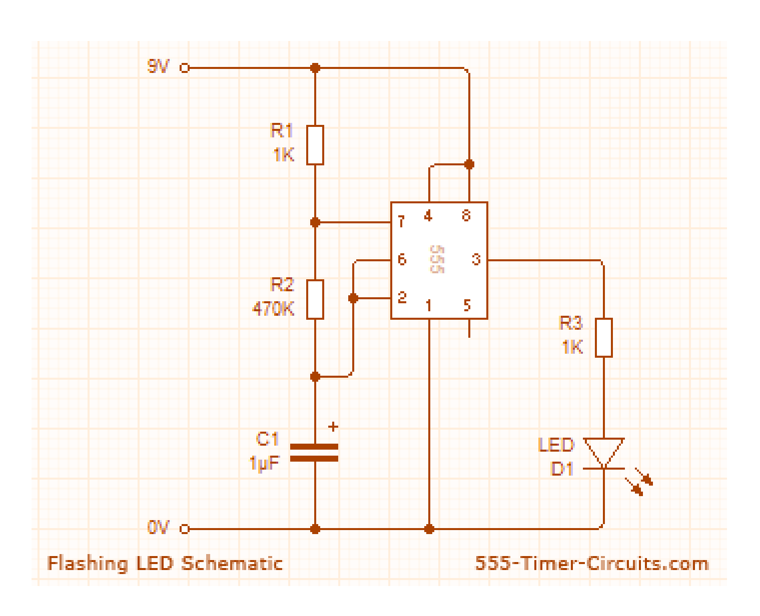
ايضا يمكن استخدام "sinking" و "sourcing" مع بعضهم البعض بحيث يمكن تشغيل جهازين وإيقاف تشغيلهما بالتناوب.




يمكن أن تكون الأجهزة أي شيء يمكن تشغيله وإيقاف تشغيله ، مثل مصابيح LED أو المصابيح أو المرحلات أو المحركات أو المغانط الكهربائية. يجب أن تكون هذه الأجهزة متصلة بالمخرج بطرق مختلفة نظرًا لأن مخرج 555 يمكن أن يؤدي فقط source او sink بتيار يصل إلى 200 ميلي أمبير , تأكد من أن مصدر الطاقة الخاص بك يمكنه توفير تيار كافي لكل من الجهاز و 555 ، وإلا سيتأثر توقيت 555.

## Flashing LED Circuit

دائرة تضيء لمبة LED وإيقاف تشغيلها , تستخدم هذه الدائرة جهاز ضبط الوقت 555 في وضع التشغيل Astable الذي يولد خرج مستمر عبر Pin 3 على شكل موجة مربعة. هذا يتحول الصمام (D1) وخارجها. يتم تعيين السرعة التي يتم فيها تشغيل وإيقاف LED D1 بقيم R1 و R2.




مواصفات القطع المستخدمة :

1x - NE555 Bipolar Timer

1x - LED (Red)

1x - 470K Resistor (1/4W)

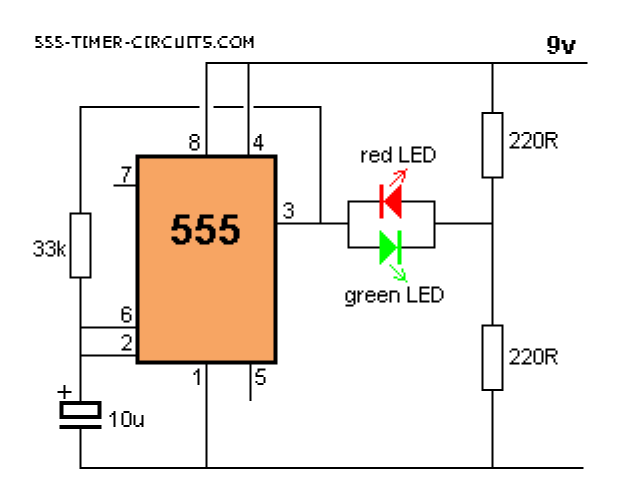
2x - 1K Resistor (1/4W)

1x - 1µF Electrolytic Capacitor (16V)

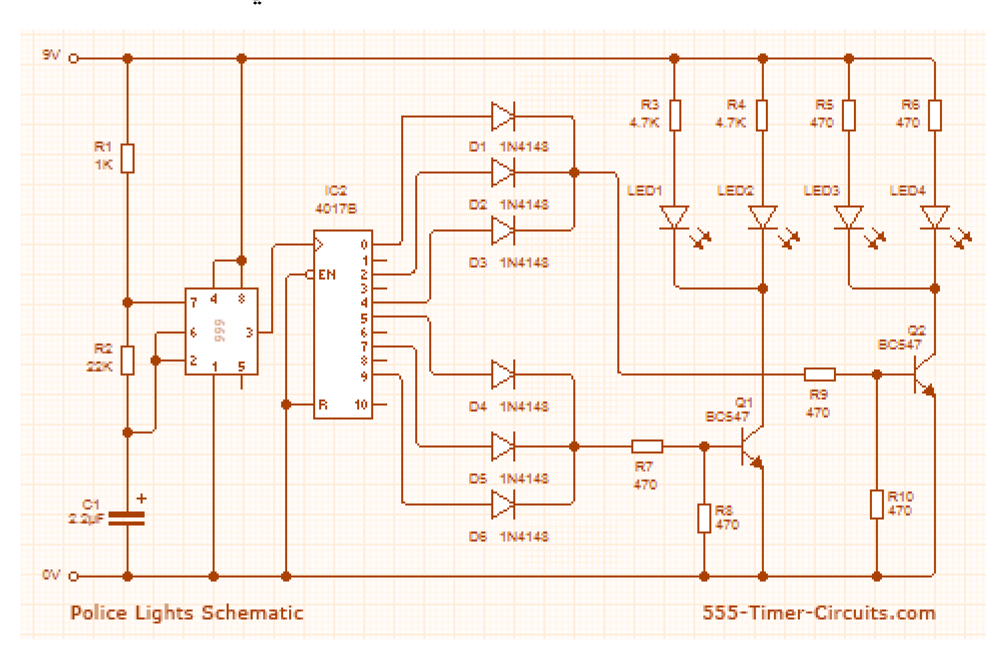
x - 9V Voltage Battery1

# BI-POLAR LED DRIVER Circuit

هذه الدائرة تقوم بعمل تومبض بين مصباحين LED مثل اضواء الشرطة :




وهذه الدائرة POLICE LIGHTS Circuit المستخدمة في اجهزة الشرطة:



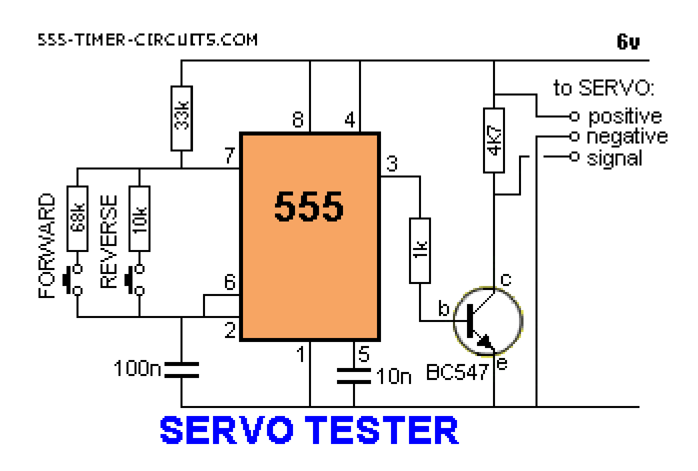
هذه الدائرة تومض لمبات LED اليسرى 3 مرات ثم LEDs اليمنى 3 مرات ، ثم يكرر.

مواصفات القطع المستخدمة :

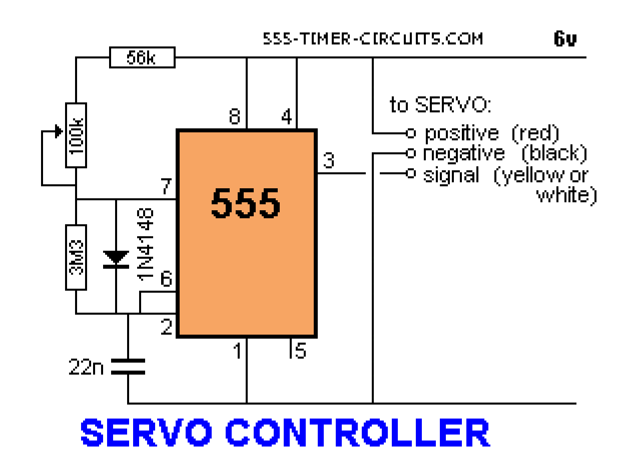
1x - NE555 Bipolar Timer
1x - 4017 Decoded Decade
6x - 1N4148 Diode
1x - 1K Resistor (1/4W)
1x - 22K Resistor (1/4W)
2x - 4.7K Resistor (1/4W)
6x - 470 Resistor (1/4W)
1x - 2.2F Electrolytic Capacitor (16V)
2x - BC547 NPN Transistor
2x - LED (Blue)
2x - LED (Red)
1x - 9V Voltage Battery

# SERVO TESTER Circuit

يمكن استخدام هذه الدائرة لتدوير السيرفو يدويًا في اتجاه عقارب الساعة وعكس عقارب الساعة. عن طريق الضغط على الزر الأمامي أو العكسي لفترة قصيرة من الوقت يمكنك التحكم في دوران السيرفو يدور المحور حوالي 180 درجة فقط :

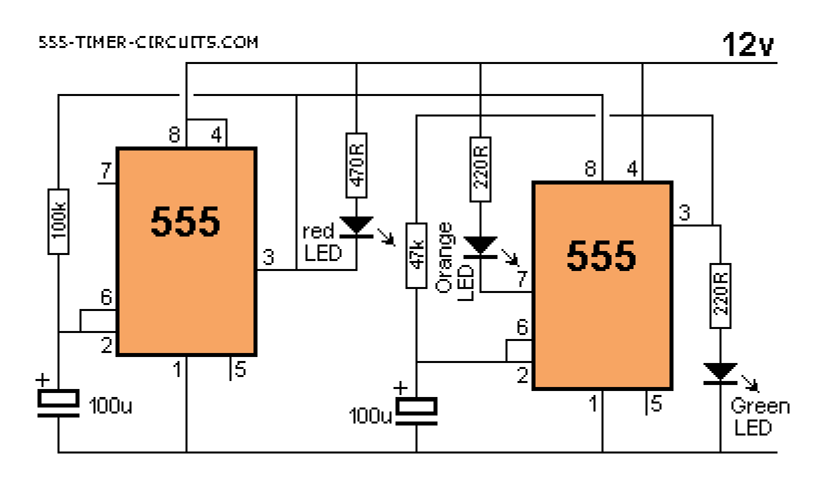



يمكن استخدام مقاومة متغيرة للتحكم في موضع الماكينة باستخدام الدائرة التالية. وتنتج نبضة موجبة بين حوالي 0 مليثانية و 2.1 ميلي ثانية. فترة إيقاف التشغيل بين النبضات هي حوالي 40 مللي ثانية. يمكن تقصير هذا عن طريق تقليل قيمة المقاوم M33.






سوف نقوم بتصميم دائرة الكترونية لمشروع اشارة مرور مكونة من 3 مصابيح LED باستخدام اثنين من الدائرة المتكاملة 555 انظر المخطط التالي يبين تسلسل الاضاءة :




يظهر المخطط تسلسل الإضاءة وهذا يتبع المعايير الأسترالية لاشارات المرور. المصباح الأحمر red LED لديه فترة متساوية عند إيقاف التشغيل ، وعند إيقاف تشغيله ، فإن أول 555 يسلم الطاقة إلى 555 الثاني. وهذا ينير مصباح LED الأخضر ، ثم حالة التغيير 555 الثانية لإيقاف تشغيل LED الأخضر وتشغيل مصباح LED البرتقالي لفترة قصيرة من الوقت قبل أول حالة تغيير 555 لإيقاف تشغيل 555 ثانية وتشغيل الصمام الأحمر. هناك حاجة لجهد إمداد من 9 فولت إلى 12 فولت لأن الثانية 555 يتلقى إمدادات أقل بنحو 2 فولت من السكك الحديدية. تُظهر هذه الدائرة أيضًا كيفية توصيل مصابيح LED عالية ومنخفضة إلى 555 وإيقاف تشغيل 555 بالتحكم في الإمداد بالدبوس 8. لن يعمل توصيل مصابيح LED العالية والمنخفضة بالدبوس 3 ، وبما أن الدبوس 7 في مرحلة مع دبوس 3 ، يمكن استخدامه للاستفادة في هذا التصميم ورؤية كيف تعمل الدائرة من هذا الموقع الالكتروني :

www.555-timer-circuits.com/traffic-lights.html

# 555 TIMER BASICS – MONOSTABLE MODE

يمكن أن يكون الموقت 555 هو الشريحة الأكثر شيوعًا المستخدمة في مشاريع الإلكترونيات اليدوية لأنها صغيرة وغير مكلفة ومفيدة للغاية.

ويعتبر جهاز توقيت لأنه يمكن أن يخرج نبضات التيار الكهربائي لفترة زمنية محددة. على سبيل المثال ، يمكن استخدامه لإيقاف مؤشر LED لمدة 5 ثوانٍ بالضبط بعد الضغط على زر. يمكن أيضًا تشغيل وميض LED وإيقاف تشغيله ، أو توليد نبضات تردد أعلى ستصدر صوتًا عند توصيله بالسماعات.

في هذا المثال سندرس الأوضاع الثلاثة المختلفة لجهاز ضبط الوقت 555 - monostable، bistable، و astable كل وضع له خصائص مختلفة تحدد كيفية إخراج مؤقت 555 .

قبل البدء اريد ان اذكر بعض الاستخدامات الصناعية لهذه الشريحة :

- السيارات
- الاتصالات
- الحواسيب
- الطاقة والاضاءة
- صناعة الاجهزة
- الفضاء
- الحماية والاغراض الطبية

في الوضع الأحادي (MONOSTABLE MODE) ، يخرج المؤقت 555 نبضة واحدة من التيار لفترة زمنية معينة. يشار إلى ذلك أحيانًا بنبض واحد.

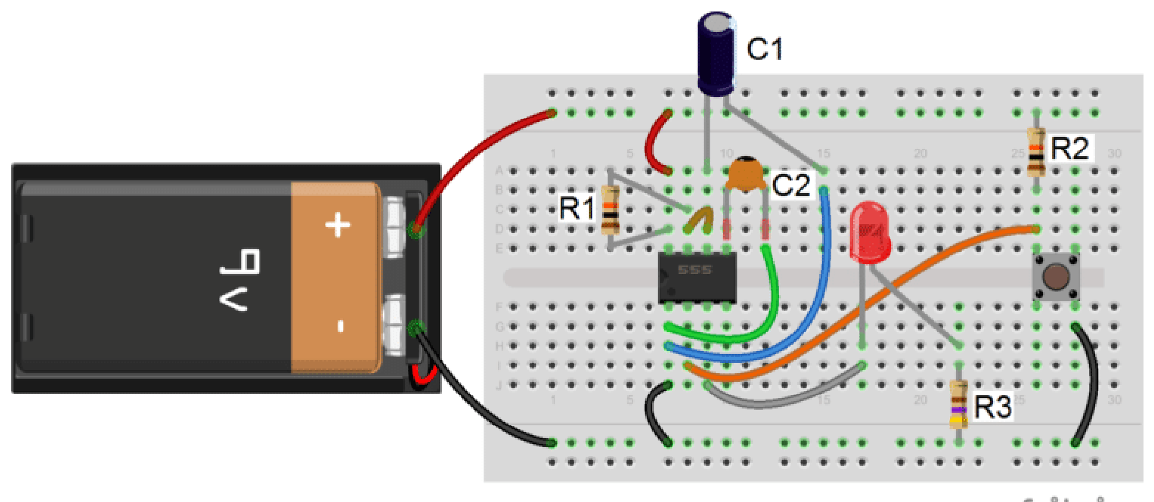
مثال على ذلك من خلال مؤشر LED وزر الضغط. بضغطة واحدة على الزر ، سوف تضيء LED ، ثم تنطفئ تلقائيًا بعد فترة زمنية محددة مسبقًا. يعتمد وقت بقاء المصباح على قيم المقاوم والمكثف المتصل بجهاز ضبط الوقت 555. يمكن حساب الوقت من المعادلة:

$$t = 1.1 \times R \times C$$

عندما يكون t هو طول المخرجات الكهربائية بالثواني ، فإن R هي مقاومة المقاوم في أوم ، و C هي سعة المكثف في فاراد.

كما ترون من المعادلة ، يمكن زيادة زمن الإخراج الكهربائي باستخدام قيم أكبر ل المقاومة أو المكثف. والعكس صحيح يمكنك الحصول على نبض إخراج أقصر مع قيم المقاومة أو مكثف أصغر.

لمراقبة الوضع الأحادي لجهاز ضبط الوقت 555 ، فلننشئ مؤقتًا بسيطًا يعمل مرة واحدة لإيقاف تشغيل مؤشر LED بعد فترة زمنية محددة. استخدم المخطط أدناه لتوصيل الدائرة:




R1: 10K Ohms

R2: 10K Ohms

R3: 470 Ohms

C1: 470 µF

C2: 0.01 µF

في هذه الدائرة ، بعد الضغط على الزر مرة واحدة ، سوف تضيء LED ثم تنطفئ بعد حوالي 5 ثوان. تحدد قيم R1 و C1 مدة بقاء LED فيما يلي :

$$t = 1.1 \times R1 \times C1$$
$$= 1.1 \times 10000\ \Omega \times 0.00047\ F$$
$$= 5.17\ seconds$$

درسنا سابقا ما يلي :

دبوس 1 - الأرض: متصلة V0

الدبوس 2 - الزناد: يتم تشغيل الإخراج عندما تنخفض الجهد الكهربائي الموفر لها عن 3/1 من.Vcc

دبوس 3 - الإخراج: إخراج ما يصل إلى 200 مللي أمبير من التيار عند حوالي 1.5 فولت.

دبوس 4 - إعادة تعيين: إعادة ضبط عملية توقيت الإخراج عندما يكون متصلاً بالأرض (0) فولت.

الدبوس 5 - التحكم: يتحكم في خرج التوقيت بشكل مستقل عن دائرة RC عندما يزيد الجهد الموفر لها عن Vcc. 3/2 في حالة عدم الاستخدام ، يتم توصيله عادةً بالأرض عبر مكثف 0.01 لمنع التقلبات في توقيت دائرة.RC

دبوس 6 - العتبة: إيقاف تشغيل الإخراج عندما يصل الجهد المورد إلى فوق (2/3) Vcc.

الدبوس 7 - التفريغ: عندما يكون الجهد الناتج منخفضًا ، فإنه يقوم بتصريف المكثف الموجود في دائرة RC إلى الأرض.

دبوس VCC 8 - (امدادات التيار الكهربائي): يمكن أن تتراوح من V 4.5 إلى 15 V.

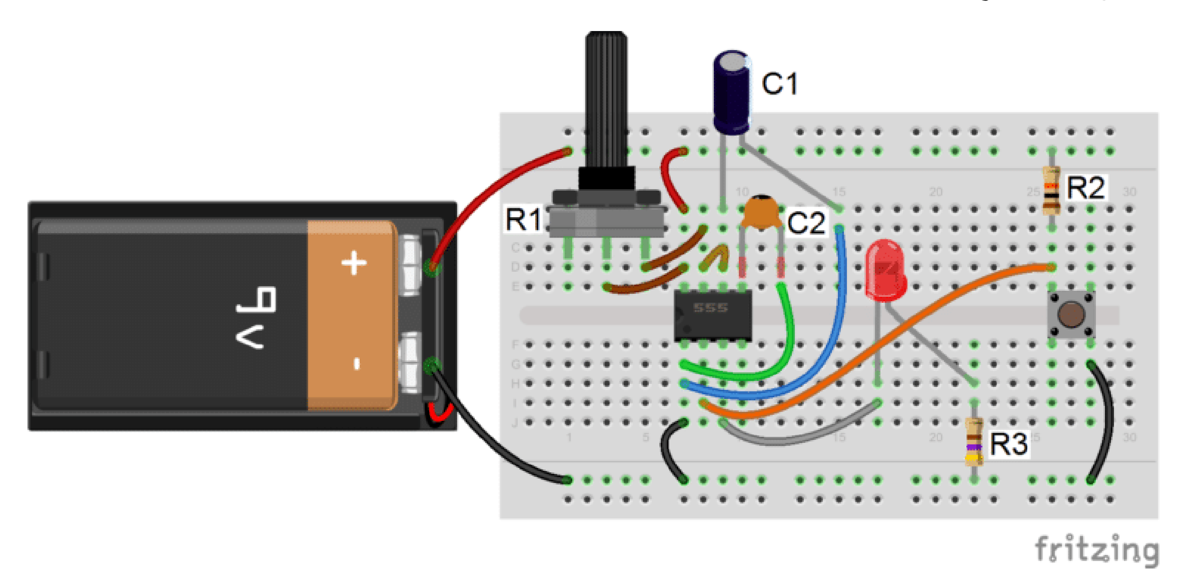
قبل الضغط على الزر ، يكون الجهد عند دبوس الزناد عاليًا. كلما كان الجهد الزناد دبوس عالية ، يسمح دبوس التصريف التدفق إلى الأرض .

عند الضغط على الزر ، ينخفض الجهد عند دبوس الزناد. عند انخفاض جهد دبوس الزناد ، يتم تشغيل دبوس الخرج. في نفس الوقت ، يوقف دبوس التفريغ تدفق التيار من C1 إلى الأرض ، مما يسمح له بالشحن.

يستغرق C1 وقتًا لشحنه ، وعلى الرغم من أن الجهد عبره أقل من (2/3) Vcc ، فإن دبوس العتبة يبقى منخفضًا حتى يظل دبوس الخرج قيد التشغيل. عندما تتراكم الشحنة أخيرًا بما يكفي لجعل الجهد الكهربي عبر C1 أكبر من 3/2 Vcc ، فإن دبوس العتبة يوقف تشغيل دبوس الخرج. في نفس الوقت ، يتم تشغيل دبوس التفريغ مرة أخرى ويمنع المكثف من الشحن حتى يتم الضغط على الزر مرة أخرى.

إن طول الوقت الذي يظل فيه مصباح LED مضيئ هو وظيفة الوقت الذي يستغرقه المكثف ليتم شحنه إلى (2/3) Vcc. يتم تحديده أيضًا بواسطة R1 ، لأن المقاوم يمنع تدفق التيار إلى المكثف وبالتالي يزيد من الوقت الذي يستغرقه الجهد للوصول إلى (2/3) Vcc.

من الطرق الجيدة لمراقبة اعتماد الوقت على المقاومة في هذه الدائرة استبدال R1 بمقاومة متغيرة :



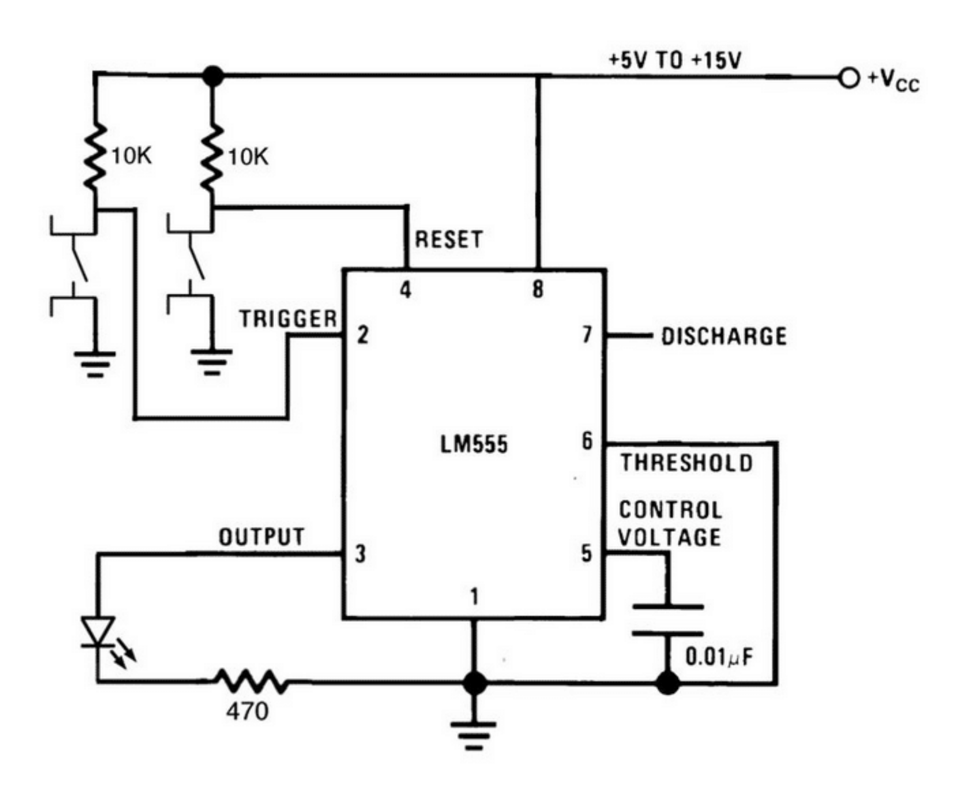

إذا قمت بضبط مقياس الجهد ، سترى أن المصباح يبدأ في الوميض بشكل أسرع أو أبطأ حسب قيمة المقاومة .

يُعرف الموقت 555 في الوضع (BISTABLE) أيضًا بدائرة انعكاس ( flip-flop circuit) . تتناوب الدائرة المتقلبة بين حالتين مستقرتين ، وفي هذه الحالة يكون خرج التيار الكهربائي من دبوس الخرج.

على عكس الوضع الأحادي والأوضاع المستقرة ، لا يحتاج هذا الوضع إلى مقاومة ومكثف لضبط توقيت الدائرة. في الواقع لا يوجد توقيت في هذه الدائرة !

هناك حالتان ثابتتان فقط (تشغيلها وإيقافها) يتم التحكم فيهما مباشرةً بواسطة دبوس الزناد ودبوس إعادة الضبط (trigger pin and reset pin) .

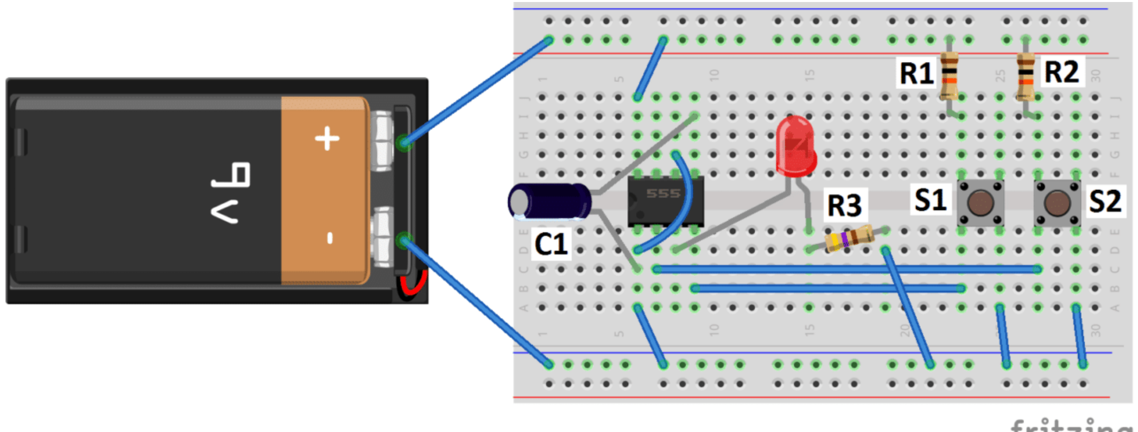
سأوضح هذا الوضع لجهاز ضبط الوقت 555 مع مؤشر LED وأزرار الضغط المتصلة بدبوس الزناد ودبوس إعادة تعيين. يؤدي الضغط على زر المشغل مرة واحدة إلى تشغيل مصباح LED واستمراره. الضغط على زر إعادة الضبط سيجعل الصمام ينطفئ ويظل في وضع إيقاف التشغيل.



الضغط على زر المشغل يسمح للتيار بالتدفق من Vcc إلى الأرض ، مما يؤدي إلى انخفاض الجهد عند دبوس الزناد. كما رأينا في وضع الوضع الأحادي (monostable mode) ، عندما يكون دبوس الزناد بجهد منخفض ، يتم تشغيل الخرج ويضيء المصباح. يستمر الخرج حتى يتجاوز الجهد عند دبوس العتبة 2/3 Vcc نظرًا لأن دبوس العتبة متصل بالأرض في هذه الدائرة ، فإنه لا يصل أبدًا إلى 2/3 Vcc، وبالتالي فإن الخرج يظل ثابتًا إلى أجل غير مسمى.

الآن إذا تم الضغط على زر إعادة الضبط ، فإن التيار الكهربائي في دبوس إعادة الضبط يتدفق إلى الأرض وينخفض مستوى الدبوس. عندما ينخفض مستوى دبوس إعادة الضبط ، يتم إيقاف تشغيل الإخراج.

لمراقبة المؤقت 555 في وضع bistable ، نفذ الدائرة التالية :




R1: 10K Ohm

R2: 10K Ohm

R3: 470 Ohm

C1: 0.01 µF

S1: Reset Button

S2: Trigger Button

الآن ، اضغط على زر المشغل (S2) مرة واحدة ، وينبغي أن يكون مؤشر LED قيد التشغيل ويستمر في التشغيل. يؤدي الضغط على زر إعادة الضبط (S1) إلى إيقاف تشغيل مؤشرLED .

## TIMER BASICS – ASTABLE MODE 555

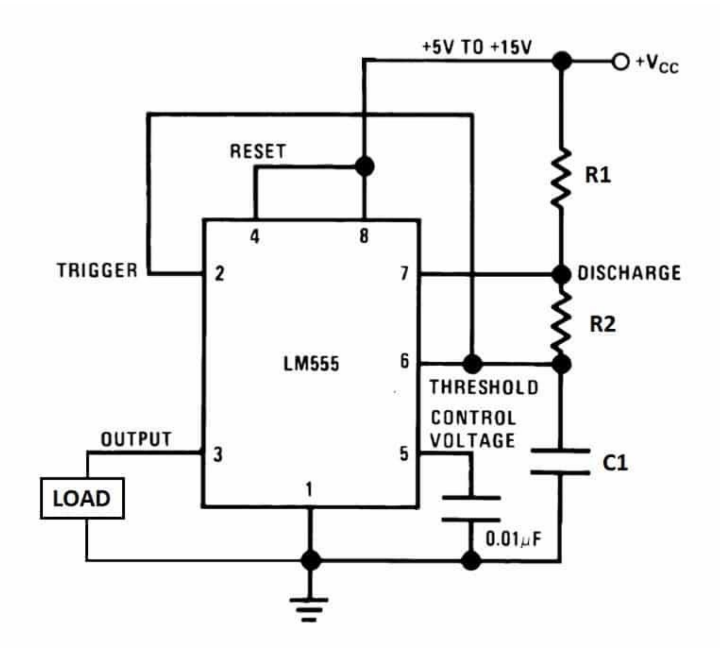
الوضع المستقر هو ما يفكر فيه معظم الناس عندما يتعلق الأمر بالمؤقت 555. في كثير من الأحيان ، عندما ترى مشروعًا مزودًا بمصابيح LED الوامضة ، يكون جهاز ضبط الوقت 555 يعمل. ولكن لديها الكثير من التطبيقات الأخرى المثيرة للاهتمام أيضًا. فمثلا، مكنه أيضًا توليد ترددات لإنتاج الصوت عند توصيل الإخراج بالسماعات. ويمكن حتى استخدامه كمحول تناظري بسيط لتحويل رقمي (ADC).

في الوضع المستقر ، يعمل الموقت 555 بمثابة مذبذب يولد موجة مربعة ويمكن ضبط تردد الموجة عن طريق تغيير قيم اثنين من المقاومات ومكثف متصل بالشريحة و ستخبرك الصيغ أدناه بطول وإيقاف دورات الإخراج بمقاومات ومكثفات مختلفة:

$$t_{on} = 0.69 \times C1 \times (R1 + R2)$$
$$t_{off} = 0.69 \times C1 \times R2$$

$t_{on}$ : *Length of high output pulse in seconds*
$t_{off}$ : *Length of low output pulse in seconds*
$R1$ : *Resistance of R1 in Ohms*
$R2$ : *Resistance of R2 in Ohms*
$C1$ : *Capacitance of C1 in Farads*

باستخدام هذه المعادلة ، يمكنك أن ترى أن زيادة قيم إما C1 أو R2 ستزيد من الوقت الذي يبقى فيه المخرج والوقت الذي يتوقف فيه. زيادة قيمه R1 سوف يطيل الوقت الذي يبقي الإخراج ON.




الدبوس 2 - الزناد: يتم تشغيل الإخراج عندما ينخفض الجهد الكهربائي الموفر لها عن 1/3 منVcc

دبوس 6 - العتبة: إيقاف تشغيل الإخراج عندما يصل الجهد الى فوق 2/3 Vcc

دبوس 7 - التفريغ: عندما يكون الجهد الناتج 0 ، فإنه يتم تصريف C1 إلى الأرض.

في الوضع المستقر ، دورات المخرج تكون (on and off) باستمرار , في المخطط أعلاه ، لاحظ أن دبوس threshold ودبوس trigger متصلان بـ C1.

هذا يجعل الجهد نفسه في دبوس الزناد(trigger) ، دبوس العتبة(threshold) و C1 .

في بداية دورة التشغيل / الإيقاف ، يكون الجهد 0 عند C1 ، ودبوس الزناد ، ودبوس العتبة, عندما يكون الجهد الزناد دبوس هو 0 ، الإخراج هو 1 ، ودبوس التفريغ هو 0

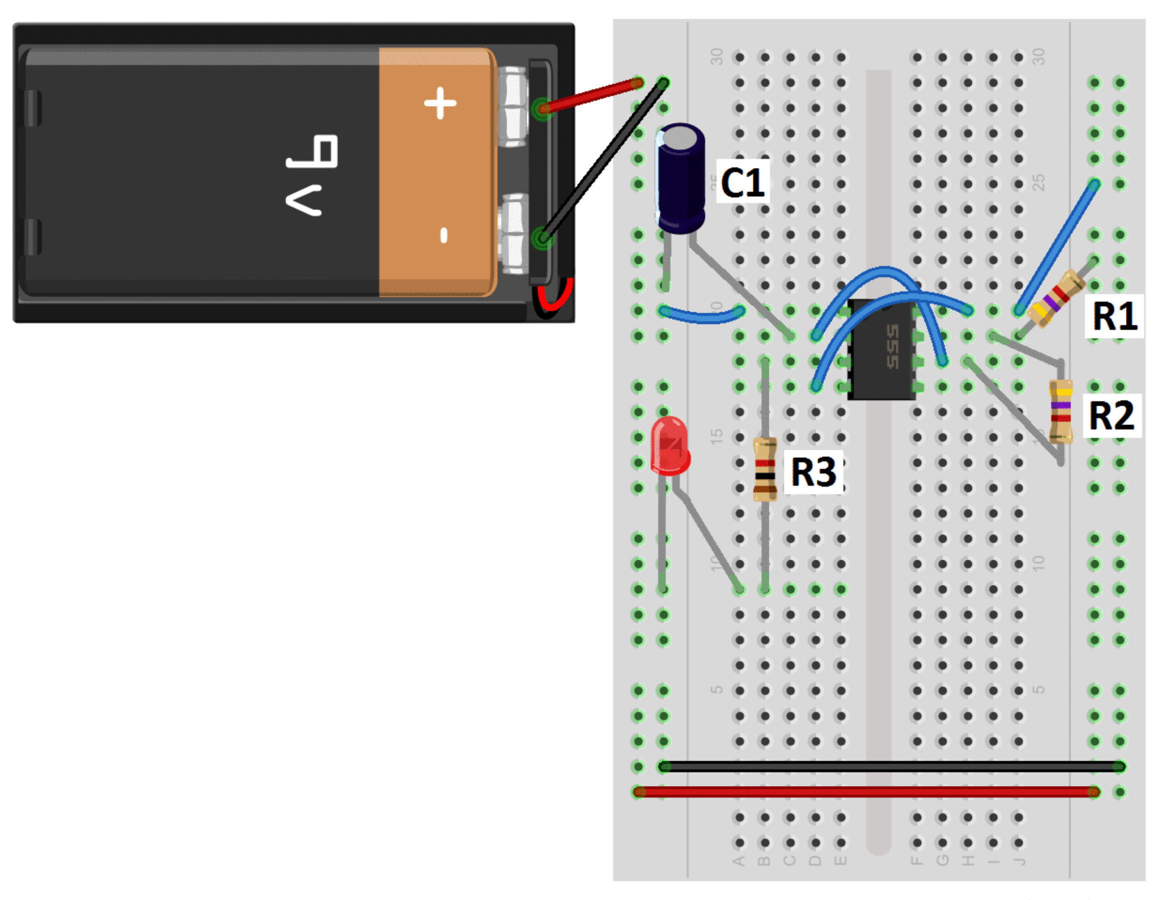
ونظرًا لأن دبوس التفريغ مطفأ ، يمكن أن يتدفق التيار عبر مقاومات R1 و R2 ، يشحن مكثف C1.

بمجرد شحن C1 إلى 2/3 Vcc، يتم إيقاف تشغيل الإخراج بواسطة دبوس العتبة. عند توقف الإخراج ، يتم تشغيل دبوس التفريغ. يسمح ذلك للشحنة المتراكمة على المكثف C1 بالتصريف إلى الأرض.

بمجرد أن ينخفض الجهد عبر C1 إلى 1/3 Vcc، يقوم دبوس الزناد بإيقاف تشغيل دبوس التفريغ ، لذلك يمكن C1 بدء الشحن مرة أخرى.

## دائرة وميض LED

لمراقبة الموقت 555 في الوضع القابل للاستقرار ، فلنقم ببناء دائرة تستخدم خرج مذبذب الموقت 555 لإنشاء ومضة LED وإيقاف تشغيلها:



R1: 4.7K Ohm resistor

R2: 4.7K Ohm resistor

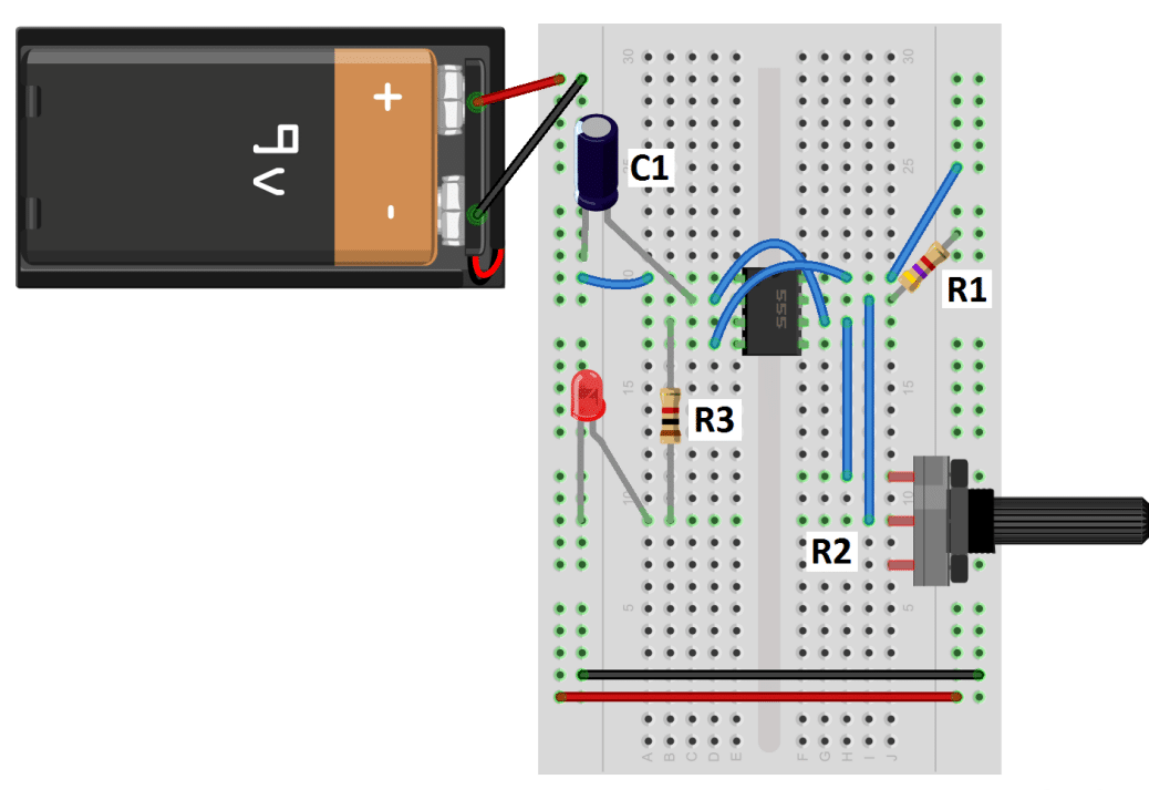
R3: 1K Ohm resistor

C1: 100 µF capacitor

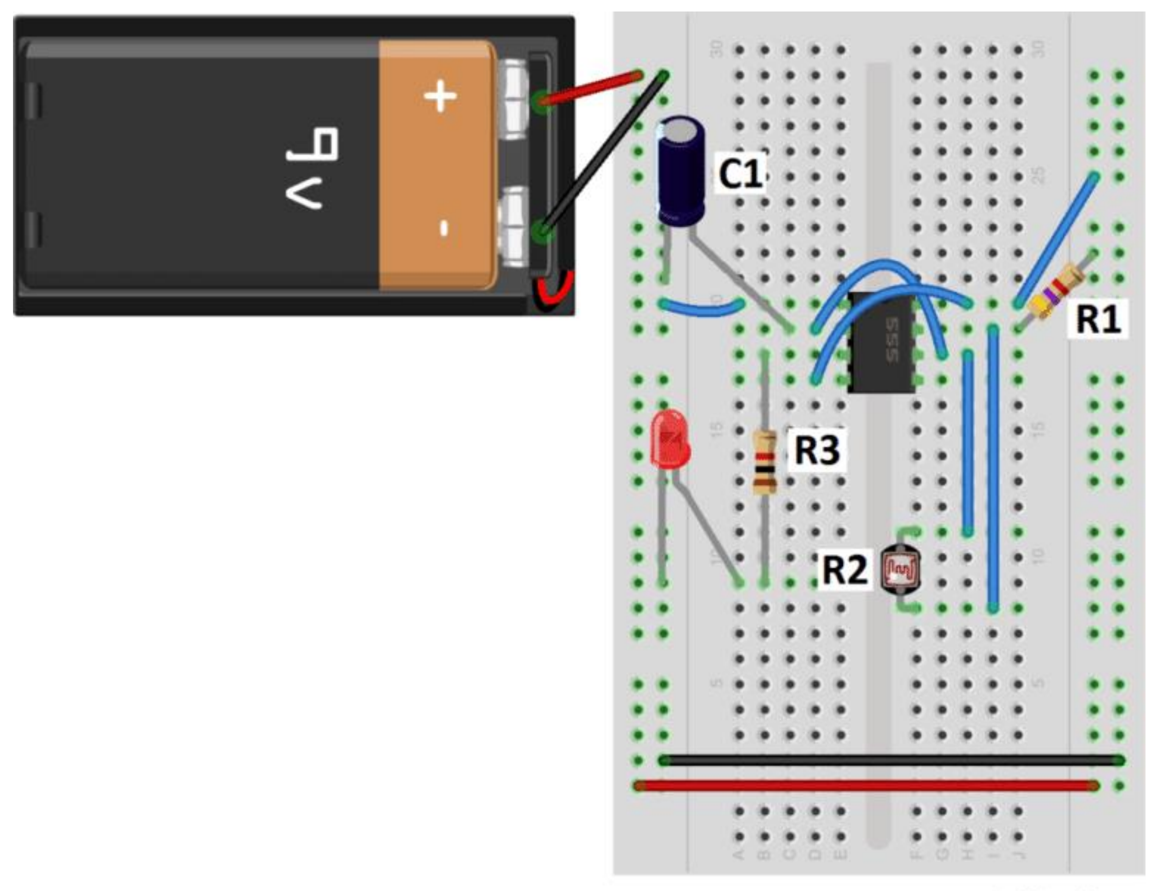
تؤثر قيم R1 و R2 و C1 على سرعة الوميض. القيم الأكبر ستجعل وميض LED أبطأ ، بينما القيم الأصغر ستجعل وميض LED أسرع.

المقاومة R3 موجود فقط لتقييد التيار إلى المصباح بحيث لا ينفجر. إذا كنت ترغب في ضبط الوامض على سرعة معينة ، يمكنك استخدام الصيغة في بداية هذا الدرس لحساب المقاومة أو السعة التي تحتاجها.

طريقة سهلة لمراقبة تأثير المقاومة على سرعة الوميض هي استخدام مقاومة متغيرة 10 K Ohm لـR2 :



بدلاً من استخدام مقاومات للتحكم في معدل الوميض ، يمكن توصيل المقاوم الضوئي (photoresistor) :




R1: 4.7K Ohm resistor

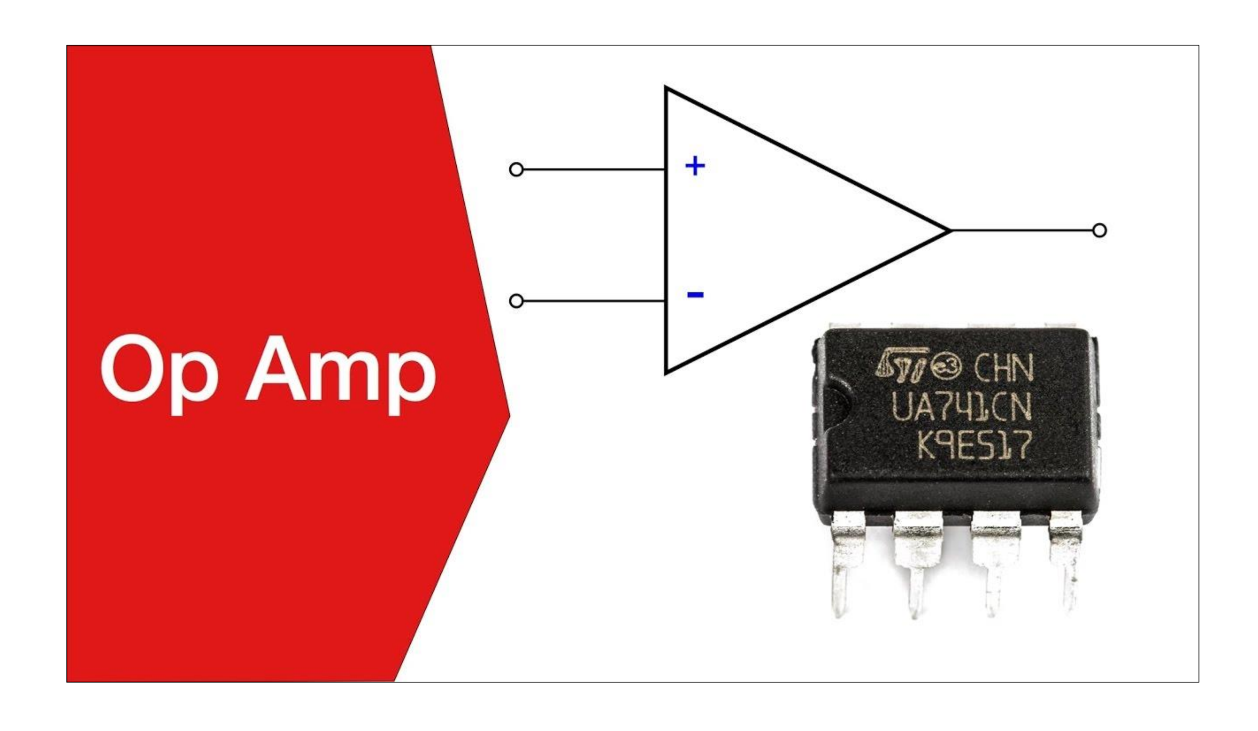
R2: Photoresistor

R3: 1K Ohm resistor

C1: 100 µF capacitor

تتناقص مقاومة photoresistor مع زيادة تسليط الضوء عليه ، وبالتالي سوف تومض LED بسرعة أكبر عندما تتعرض لمزيد من الضوء.

## مضخم العمليات Amplifier




مضخم العمليات هو المصطلح العام المستخدم لوصف الدائرة التي تنتج وتزيد من قيمة إشارة الإدخال اذا المضخم هو جهاز أو دائرة إلكترونية تستخدم لزيادة حجم الإشارة المطبقة على مدخلاتها لتكبير وتضخيم الجهود المستمرة والاشارات المتناوبة. ومع ذلك ، ليست جميع دارات المضخم هي نفسها كما يتم تصنيفها وفقًا لتكوين داراتها وأنماط وحالات تشغيلها.

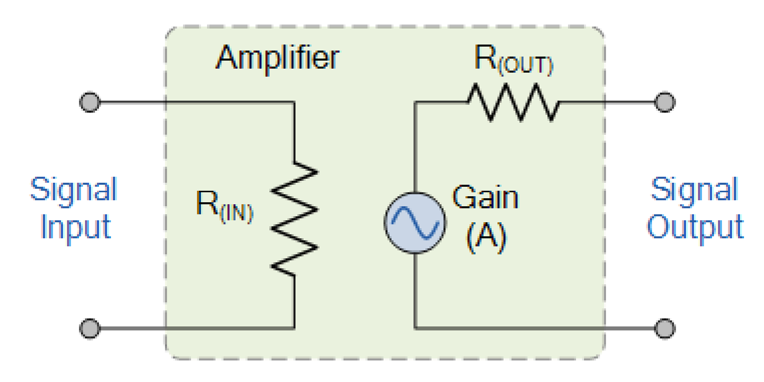
في الإلكترونيات ، تكون مضخمات الإشارة الصغيرة هي الأجهزة المستخدمة بشكل شائع ، حيث إنها تمتلك القدرة على تضخيم إشارة دخل صغيرة نسبيًا ، على سبيل المثال جهاز مكبر الصوت وبعض التطبيقات الاخرى مثل استخدامه في دوائر المقارانات والمنظمات

يتم تصنيف مكبرات العمليات كما يلي :

| Type of Signal | Type of Configuration | Classification | Frequency of Operation |
|---|---|---|---|
| Small Signal | Common Emitter | Class A Amplifier | Direct Current (DC) |
| Large Signal | Common Base | Class B Amplifier | Audio Frequencies (AF) |
| | Common Collector | Class AB Amplifier | Radio Frequencies (RF) |
| | | Class C Amplifier | VHF, UHF and SHF Frequencies |

يمكن اعتبار المكبرات بمثابة مربع أو كتلة بسيطة تحتوي على جهاز التضخيم ، مثل ترانزستور ثنائي القطب أو ترانزستور تأثير المجال ، الذي يحتوي على محطتي إدخال ومحطتين للإخراج مع إشارة خرج أكبر بكثير من إشارة الدخل كما تم بسبب تضخيمها.

سيكون لمكبر الإشارة المثالي ثلاثة خصائص رئيسية: مقاومة الإدخال (RIN)، مقاومة الخرج (ROUT) والتضخيم المعروف عادة باسم Gain أو (A) بغض النظر عن مدى تعقيد دارة المكبر ، لكن من الممكن استخدام نموذج مكبر للصوت عام لإظهار العلاقة بين هذه الخصائص الثلاث كما هو موضح بالصورة التالية :




يُعرف الفرق المتضخم بين إشارات الدخل والإخراج باسم كسب المكبر. الكسب هو في الأساس قياس لمقدار "تضخيم" إشارة الدخل. على سبيل المثال ، إذا كانت لدينا إشارة دخل تبلغ 1 فولت وناتج يبلغ 50 فولت ، فإن كسب مكبر الصوت سيكون "50". وبعبارة أخرى ، تم زيادة إشارة الإدخال بعامل مقداره 50. وتسمى هذه الزيادة Gain.

كسب مكبرات الصوت هو ببساطة نسبة المخرجات مقسوما على المدخلات. لا تملك Gain أي وحدات فيزيائية لها ، ولكن في الإلكترونيات عادة ما يتم إعطاء الرمز "A" للتضخيم. ثم يتم ببساطة حساب مكسب الصوت على أنه ،

"إشارة خرج مقسومة على إشارة الدخل".

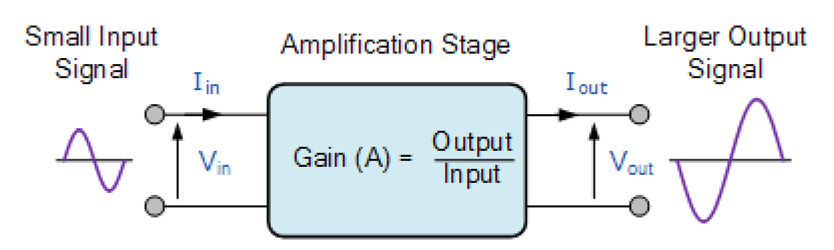
يمكن القول بأن كسب المضخم هي العلاقة الموجودة بين الإشارة المقاسة عند الخرج مع الإشارة المقاسة عند المدخل. هناك ثلاثة أنواع مختلفة من مكاسب مكبر الصوت والتي يمكن قياسها وهي:

Voltage Gain ( Av )

Current Gain ( Ai )

Power Gain ( Ap )

اعتماداً على الكمية التي يتم قياسها بأمثلة من هذه الأنواع المختلفة من الكسب موضحة أدناه :



$$Voltage\,Gain\,(A_v) = \frac{Output\ Voltage}{Input\ Voltage} = \frac{Vout}{Vin}$$

$$Current\,Gain\,(A_i) = \frac{Output\ Current}{Input\ Current} = \frac{Iout}{Iin}$$

$$Power\,Gain\,(A_p) = A_v \ x \ A_i$$

لاحظ أنه بالنسبة إلى Power Gain ، يمكنك أيضًا تقسيم الطاقة التي تم الحصول عليها عند الإخراج باستخدام الطاقة التي تم الحصول عليها عند الإدخال. أيضاً عند حساب كسب المكبّر ، تُستخدم v و i و p للإشارة إلى نوع كسب الإشارة المستخدم.

يمكن أيضاً التعبير عن كسب الطاقة (Ap) أو مستوى قدرة المضخم في الديسيبل (dB). إن Bel هو وحدة لوغاريتمية (قاعدة 10) للقياس لا تحتوي على وحدات. بما أن Bel هي وحدة قياس كبيرة جدا ، فهي مسبوقة ب deci ، مما يجعلها ديسيبل بدلا من الديسيبل واحد واحد عشر (10/1) من Bel. لحساب كسب مكبر الصوت في ديسيبل أو ديسيبل ، يمكننا استخدام العلاقات التالية :

- Voltage Gain in dB: $a_v$ = 20*log(Av)
- Current Gain in dB: $a_i$ = 20*log(Ai)
- Power Gain in dB: $a_p$ = 10*log(Ap)

إن كسب طاقة التيار المستمر لمضخم يساوي عشرة أضعاف log الشائع لنسبة المخرجات إلى المدخلات ، حيث تكون المكاسب الفولطية والتيارية 20 مرة من log العام للنسبة. ومع ذلك ، فإن dB20 ليس ضعف الطاقة بمقدار dB10 بسبب مقياس log.

كذلك ، تمثل القيمة الموجبة للديسيبل dB كسباً بينما تمثل القيمة السالبة dB خسارة في التكبير. على سبيل المثال ، يشير كسب مكبر للصوت لـ + DB3 إلى أن إشارة خرج مضخمات الصوت قد "تضاعفت" ، (x2) في حين أن كسب مضخم قدره -3 dB يشير إلى أن الإشارة قد "انخفضت إلى النصف" أو (x0.5) أو بمعنى آخر خسارة .

وتسمى نقطة -3 ديسيبل من المكبر نقطة نصف الطاقة half-power point التي هي -3 ديسيبل أسفل من الحد الأقصى ، مع الأخذ في الاعتبار dB0 قيمة اللحد الاقصى

مثال : تحديد الجهد والتيار وكسب الطاقة لمكبر الذي يحتوي على إشارة دخل mA1 عند mV10 وإشارة خرج المقابلة من mA10 عند V1. أيضا ، وضح المكاسب الثلاث في ديسيبل :

Amplifier Gains:

$$A_V = \frac{\text{Output Voltage}}{\text{Input Voltage}} = \frac{1}{0.01} = 100$$

$$A_i = \frac{\text{Output Current}}{\text{Input Current}} = \frac{10}{1} = 10$$

$$A_p = A_V \times A_i = 100 \times 10 = 1{,}000$$

Amplifier Gains given in Decibels (dB):

$$a_v = 20\log A_v = 20\log 100 = 40\,dB$$

$$a_i = 20\log A_i = 20\log 10 = 20\,dB$$

$$a_p = 10\log A_p = 10\log 1000 = 30\,dB$$

عندئذ يكون للمكبر مكاسب الجهد ، (Av) 100 ، الكسب ، (Ai) 10 ومكسب القدرة 1،000 (Ap)

بشكل عام ، يمكن تقسيم مكبرات الصوت إلى نوعين مختلفين اعتمادا على القوة أو كسب الجهد. نوع يسمى مضخم الإشارة الصغيرة الذي يتضمن مكبرات صوتية مسبقة ، مضخمات أجهزة القياس ، إلخ. تضخيم إشارة صغيرة مصممة لتضخيم مستويات جهد إشارة صغيرة جداً من عدد قليل جدًا من الفولتية الصغيرة (μV) من المستشعرات أو الإشارات الصوتية.

أما النوع الآخر فيطلق عليه اسم مضخمات الإشارة الكبيرة مثل مضخمات الطاقة الصوتية أو مضخمات تحويل الطاقة. تم تصميم مضخمات الإشارة الكبيرة لتضخيم إشارات الجهد الكهربي الكبيرة أو تبديل تيارات الحمل الثقيل كما ستجد مكبرات الصوت.

يشار إلى مضخم الإشارة الصغيرة بصفة عامة على أنه مضخم "Voltage" لأنه عادة ما يقوم بتحويل جهد دخل صغير إلى جهد إخراج أكبر . في بعض الأحيان تكون دائرة المضخم مطلوبة لتشغيل محرك أو تغذية مكبرات الصوت وبالنسبة لهذه الأنواع من التطبيقات التي تحتاج فيها تيارات التحويل العالية تحتاج إلى مكبرات الصوت Power Amplifiers.

كما يوحي اسمها ، فإن المهمة الرئيسية لـ "Power Amplifiers " (المعروف أيضًا بمكبر الإشارة الضخم) ، هو توصيل الطاقة إلى الحمل ، وكما نعلم من الأعلى ، هو نتاج الجهد والتيار المطبق على تحميل مع طاقة إشارة الخرج تكون أكبر من قوة إشارة الدخل.

وبعبارة أخرى ، يضخم Power Amplifiers قوة إشارة الدخل مباشرة ، وهذا هو السبب في استخدام هذه الأنواع من دارات المضخمات في مراحل خرج مضخم لتشغيل مكبرات الصوت.

مكبر للصوت المثالي سيعطينا كفاءة بنسبة 100٪ أو قوة "IN" ستكون مساوية لطاقة "OUT". ومع ذلك ، في الواقع ، لا يمكن أن يحدث هذا أبداً حيث يتم فقدان بعض القوة في شكل حرارة وأيضاً ، يستهلك مكبر الصوت نفسه الطاقة أثناء عملية التضخيم. ثم يتم إعطاء كفاءة مكبر للصوت على النحو التالي:

$$Efficiency\ (\eta) = \frac{\text{power delivered to the load}}{\text{d.c. power taken from the supply}} = \frac{Pout}{Pin}$$

يمكننا تحديد الخصائص لمكبر للصوت المثالي من مناقشتنا السابقة فيما يتعلق بمكاسبه ، وتحديدا كسب الجهد:

يجب أن تحافظ المكبرات على نسبة كسب ثابتة بالنسبة لقيم إشارات الدخل المتغيرة.

لا يتأثر الكسب بالتكرار حيث يجب تضخيم إشارات جميع الترددات بنفس المقدار بالضبط.

لا يجب أن تضيف مكبرات ضوضاء إلى إشارة الخرج. يجب إزالة أي ضجيج موجود بالفعل في إشارة الدخل.

لا ينبغي أن تتأثر مكبرات بالتغيرات في درجة الحرارة , مما يمنح درجة حرارة جيدة ولا يتعطل .

يجب أن يبقى كسب المكبر ثابتًا على مدار فترات زمنية طويلة.

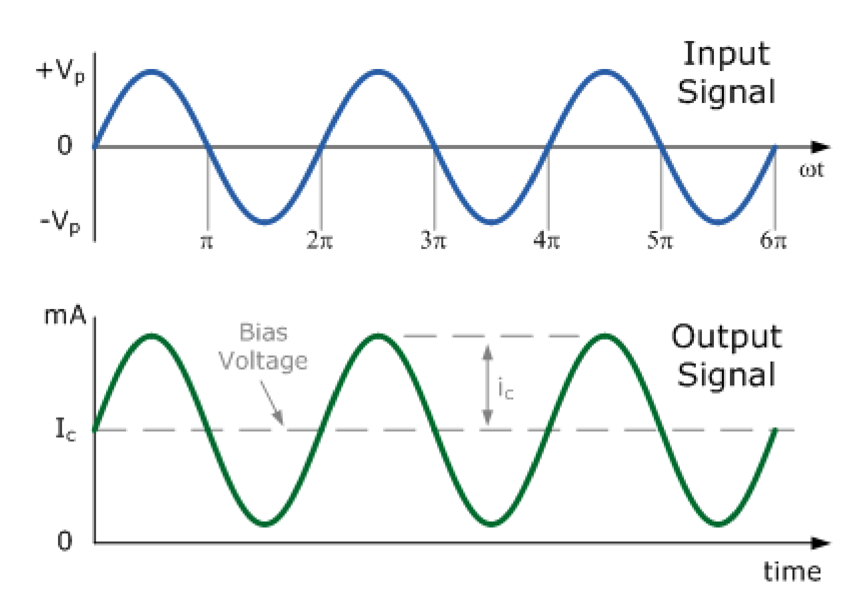
فئات المكبر الإلكتروني :

يتم تصنيف مضخم الصوت إما كجهد أو مضخم طاقة بمقارنة خصائص إشارات الدخل والإخراج من خلال قياس مقدار الوقت بالنسبة لإشارة الدخل التي يتدفق بها التيار في دائرة الخرج.

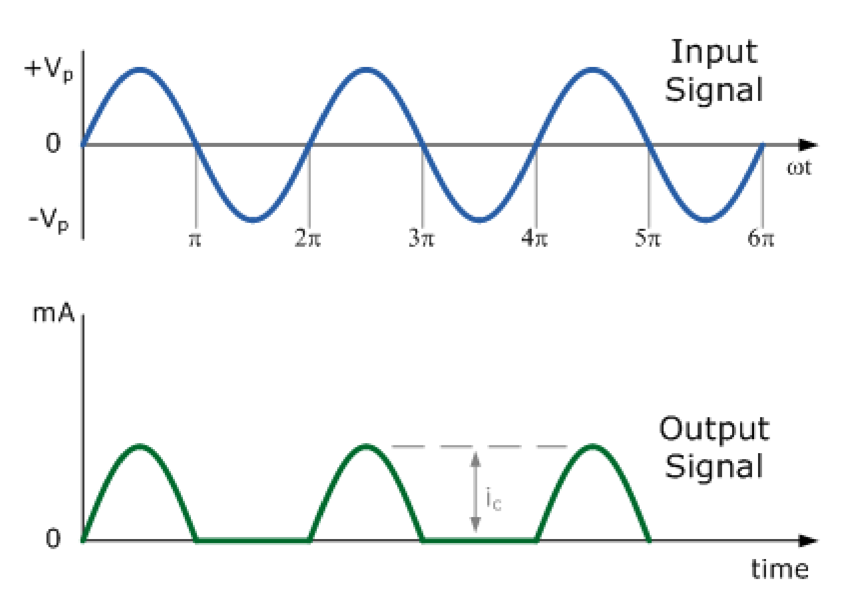
يتم تصنيف مضخمات قوة الصوت بترتيب أبجدي وفقًا لتكوينات داراتها وطريقة عملها. يتم تعيين مكبرات الصوت بواسطة فئات مختلفة من التشغيل مثل الفئة "A" ، والفئة "B" ، والفئة "C" ، والفئة "AB" ، وما إلى ذلك. تتراوح فئات المكبرات المختلفة هذه من ناتج خطي قريب ولكن بكفاءة منخفضة إلى غير الإخراج الخطي ولكن بكفاءة عالية.

لا توجد فئة واحدة من العمليات هي "أفضل" أو "أسوأ" من أي فئة أخرى فكل فئة لها استخدامها. توجد كفاءات نموذجية للتحويلات القصوى للأنواع المختلفة أكثرها استخدامًا:

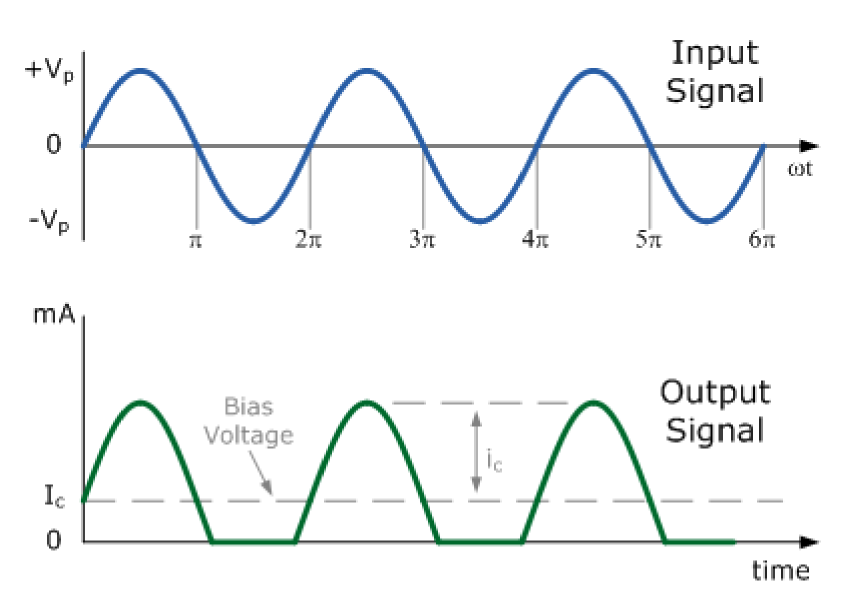
فئة A - لديها كفاءة منخفضة أقل من 40 ٪ ولكن قدرة الاستنساخ للإشارة جيدة وهي خطية.




الفئة B - لديها كفاءة منخفضة أقل من 40 ٪ الكفاءة النظرية القصوى بنسبة 70٪ تقريبًا نظرًا لأن جهاز التكبير يعمل فقط لنصف إشارة الدخل ويستهلك طاقة.



الفئة AB Amplifier - لديها تصنيف كفاءة مثل A و B ولكن تستخدم إنتاج إشارة أقل من مضخمات الفئة A.



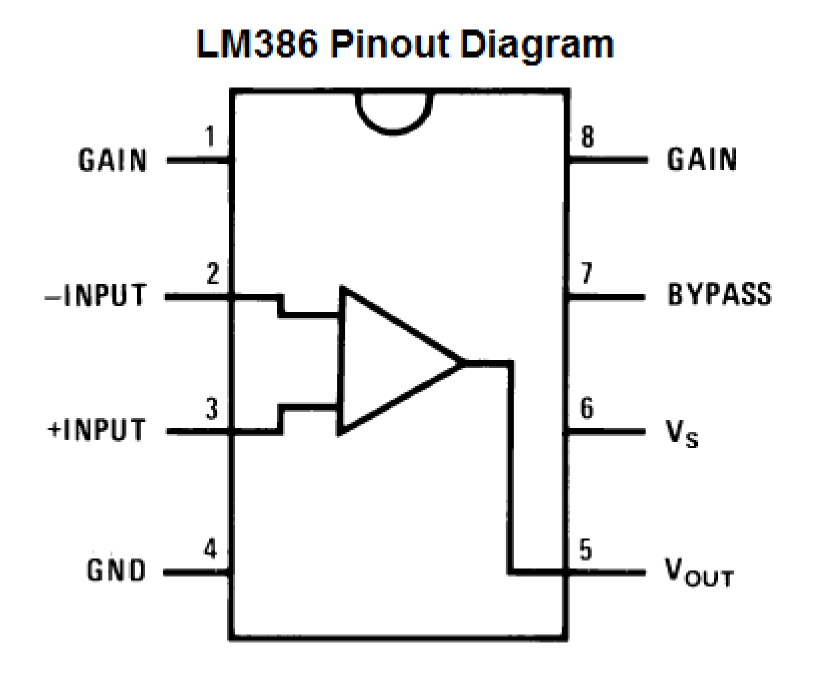

الفئة C - هو أكثر فئة مكبرة كفاءة ولكن التشوه مرتفع للغاية حيث يتم تضخيم جزء صغير فقط من إشارة الدخل ، وبالتالي فإن إشارة الخرج تحمل تشابهًا قليلًا مع إشارة الدخل, مكبرات الصوت من الفئة C لديها أسوأ إعادة إنتاج للإشارة.

# رقاقة مضخم الصوت LM386




LM386هي رقاقة مضخم صوت مشهورة للغاية تسمح للمستخدم بتضخيم الصوت , تستخدم على نطاق واسع ، بشعبية كبيرة. سوف ندرس جوانب هذه الشريحة بما في ذلك نهايات أطرافها وكيفية توصيلها بدائرة.

الرسم البياني للشريحة :




الاطراف :

الطرفين 1 و 8 : تمثلان التحكم في مكاسب المضخم (Gain) هذه هي الاطراف التي يمكنك فيها ضبط الكسب عن طريق وضع مقاومة ومكثف أو مكثف بين هذين النقطتين في هذه الدائرة سنقوم بوضع مكثف 10 µFللحصول على أعلى كسب للجهد.

الطرفين 2 و 3 : هي اطراف إشارة الدخل الصوتي. هذه هي الاطراف التي تضع فيها الصوت الذي تريد تضخيمه. الطرف 2 هي وحدة الإدخال السالبة - و 3 هي الإدخال الموجبة +.

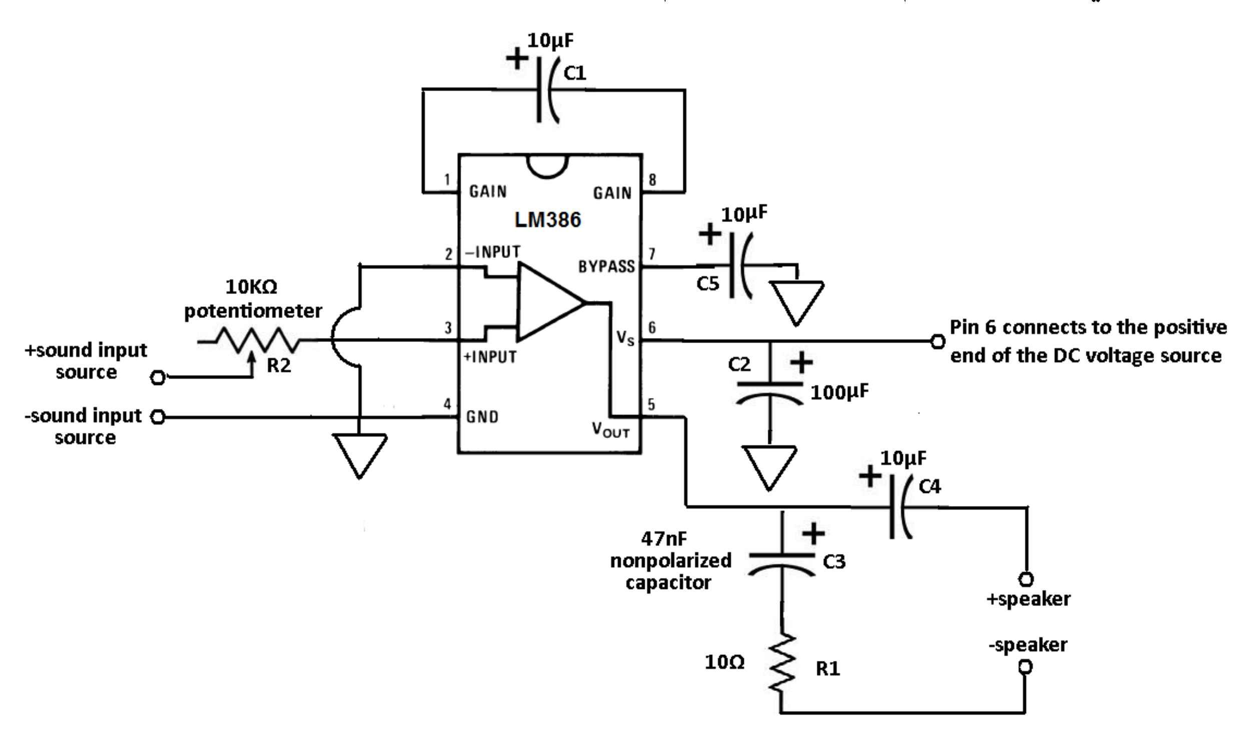
الطرف 4 : GND ground

الطرف 5 : مخرج تكبير الصوت, هذه هي المحطة التي تخرج فيها إشارة الصوت المضخمة.

الطرف 6 : التغذية الموجبة للمكبر

الطرف 7 : عادة ما يترك هذا الطرف مفتوحًا أو موصلًا بخط ألارضي. ومع ذلك ، لتحقيق استقرار أفضل ، يضاف مكثف لأن هذا يمكن أن يمنع التذبذبات في رقاقة المضخم .

فيما يلي دارة لتضخيم الصوت باستخدام LM386




المكونات الخارجية الاضافية :

C1 هو مكثف يستخدم لتحديد الكسب في المضخم, من خلال وضع مكثف هنا ، يمكننا ضبط الكسب لأعلى مستوى للحصول على أقصى قدر من الكسب ، وهو في هذه الحالة.

C2 هو مكثف تنعيم. إذا كان المصدر الكهربائي لديه أي جهد مفاجئ أو تيار مستمر ، فإن هذا المكثف يعمل على تخفيف وتنعيم الإشارة. يساعد هذا المكثف على التخلص من كل تلك التموجات ، ويستفيد المكبر من ذلك للحصول على ضوضاء أقل داخله.

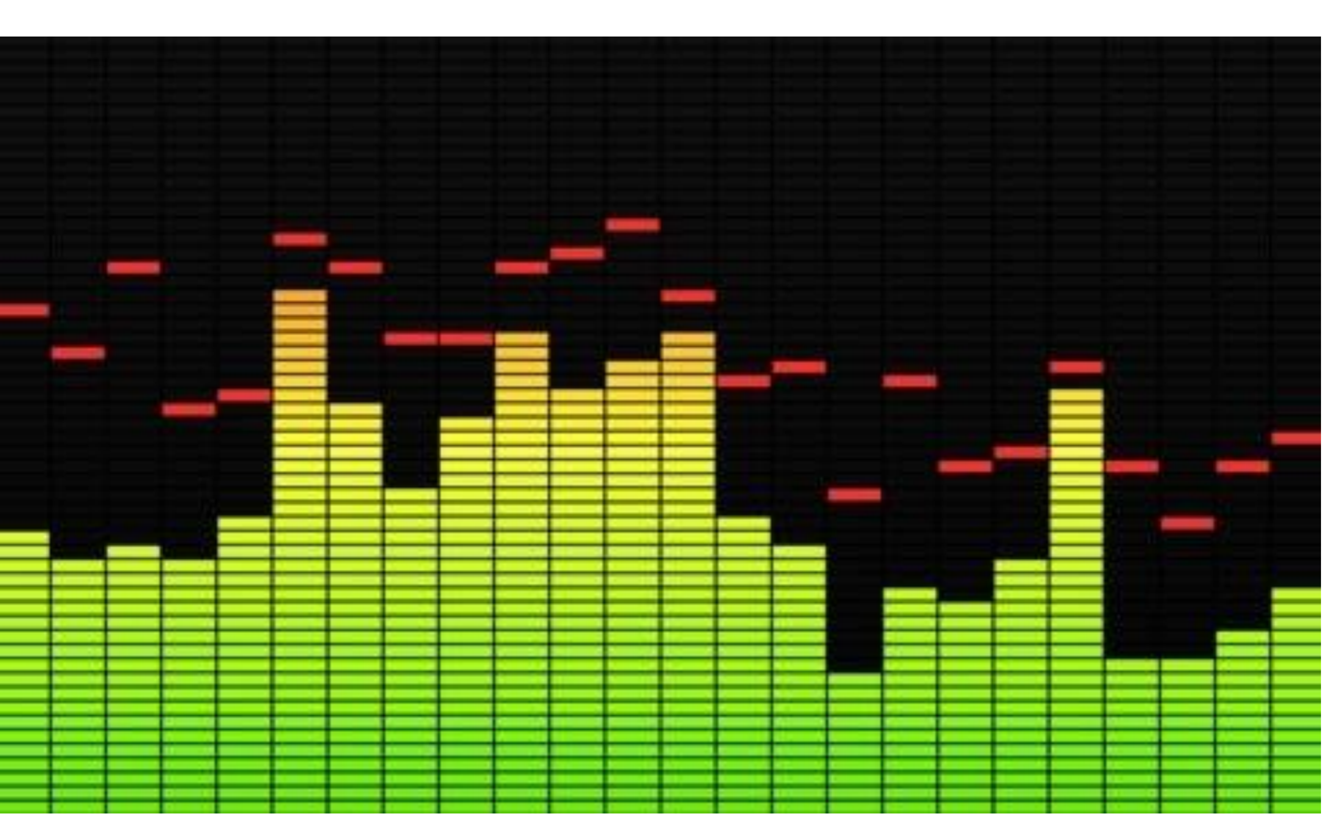
C3 يعمل كبنك للمخرجات . بحيث يملأ هذا المكثف الإلكترونات عندما يكون الطلب على التيار منخفضًا ويستنزف عندما تحدث زيادات مفاجئة للتيار.

C4 هو مكثف ربط . فهو يزيل التيار مستمر من خرج مكبر الصوت LM386 ويمرر فقط إشارة الصوت (AC).

C5 يحسن استقرار مكبر الصوت LM386 لمنع مشاكل مثل التذبذب. يمكن التذبذب تحويل الأصوات المألوفة إلى لا يمكن التعرف عليها.

جميع المكبرات تحتاج إلى الجهد DC يحتاج LM386 من أجل تشغيل ان يأخذ 4-12 فولت من جهد التيار المستمر ليعمل.

## بناء مكبر صوت رائع (مع زيادة في الصوت) باستخدام LM386



هذا المضخم الذي سوف نبنيه ليس مضخم صوت عادي "باستخدام المكونات الدنيا". لقد أضفنا مجموعة من المكثفات الإضافية لتقليل الضوضاء ، وأضفت تحكمًا في زيادة الصوت العالي لجعله يبدو أفضل. ولكن قبل البدء في البناء ، قد يكون من المفيد الحصول على المعلومات الاساسية الصغيرة أولا

تعد LM386 شريحة متعددة الاستخدامات. هناك حاجة إلى اثنين فقط من المقاومات والمكثفات لصنع مكبر صوت , وتحتوي الشريحة على خيارات للتحكم في الصوت وزيادة الصوت باستخدام المفاتيح ، ويمكن أيضًا تحويلها إلى مذبذب قادر على إخراج الموجات الجيبية أو الأمواج المربعة.

هناك ثلاثة أنواع من LM386 ، ولكل منها تصنيفات طاقة خرج مختلفة:

- LM386N-1: 0.325 Watts
- LM386N-3: 0.700 Watts
- LM386N-4: 1.00 Watts

ستعتمد طاقة الخرج الفعلية التي تحصل عليها على جهد الإمداد ومعاوقة (ممانعة) مكبر الصوت. تحتوي ورقة البيانات (Data sheet) على رسوم بيانية ستخبرك بهذه التفاصيل وايضا للحصول على مزيد من المعلومات حول طاقة الخرج وخصائص التشويه والحد الأدنى / الحد الأقصى للتقييمات , لقد استخدمت بطارية 9 فولت لإمداد الطاقة وهي تعمل بشكل رائع ، لكن يمكنك النزول إلى 4 فولت أو حتى الارتفاع الى 12 فولت.

يأخذ LM386 إشارة إدخال صوت ويزيد من إمكاناته في أي مكان من 20 إلى 200 مرة. هذا التضخيم هو ما يعرف بزيادة الجهد كما درسنا سابقا .

عند إنشاء هذا المضخم وتغير مستوى الصوت باستخدام التحكم في مستوى التضخيم والتحكم اذا الكسب هو تضخيم إمكانات الإدخال وهو أحد خصائص المضخم.

يتيح لك مستوى الصوت ضبط مستوى الصوت في نطاق التضخيم الذي حدده الكسب.

الكسب يحدد نطاق مستويات الصوت الممكنة. على سبيل المثال ، إذا تم تعيين الكسب الخاص بك على 20 ، فإن نطاق تضخيم الصوت يكون من 0 إلى 20. إذا تم تعيين الكسب الخاص بك على 200 ، فإن نطاق التضخيم يكون من 0 إلى 200 .

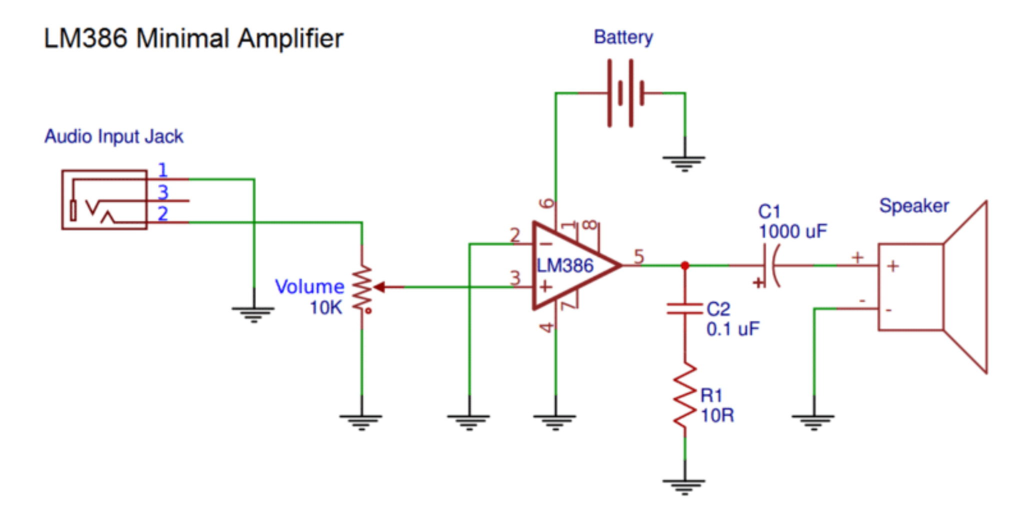
يمكن التحكم بالكسب عن طريق توصيل مكثف 10 µF بين الطرفين 1 و 8. بدون مكثف بين السنين 1 و 8 ، سيتم ضبط الكسب تلقائيا على 20.

باستخدام مكثف 10 µF ، سيتم ضبط الكسب على 200. يمكن تغيير الكسب إلى أي قيمة بين 20 و 200 عن طريق وضع مقاومة (أو مقاومة متغيرة) على التوالي مع المكثف.

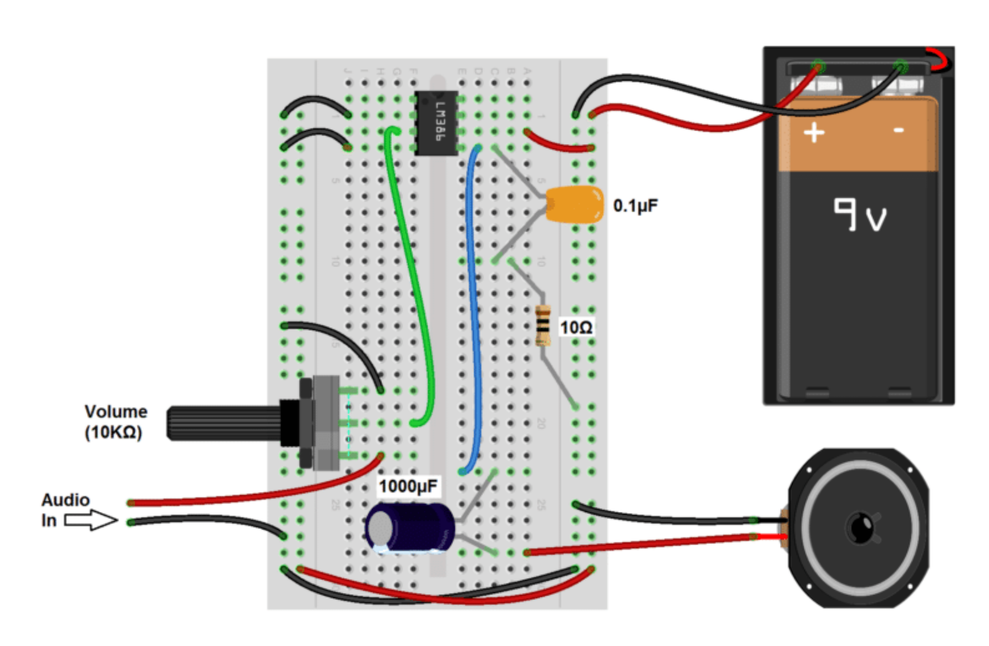
## مكبر صوت صغير LM386

الآن بعد أن لدينا القليل من المعلومات الأساسية عن LM386 ، فلنبدأ من خلاله بناء مكبر للصوت مع الحد الأدنى من المكونات اللازمة لجعله يعمل وبهذه الطريقة يمكنك مقارنتها بالمكبرات الأفضل الذي سنبنيه لاحقًا.

المخطط :




إليك كيفية توصيله إذا كنت تستخدم لوحة:

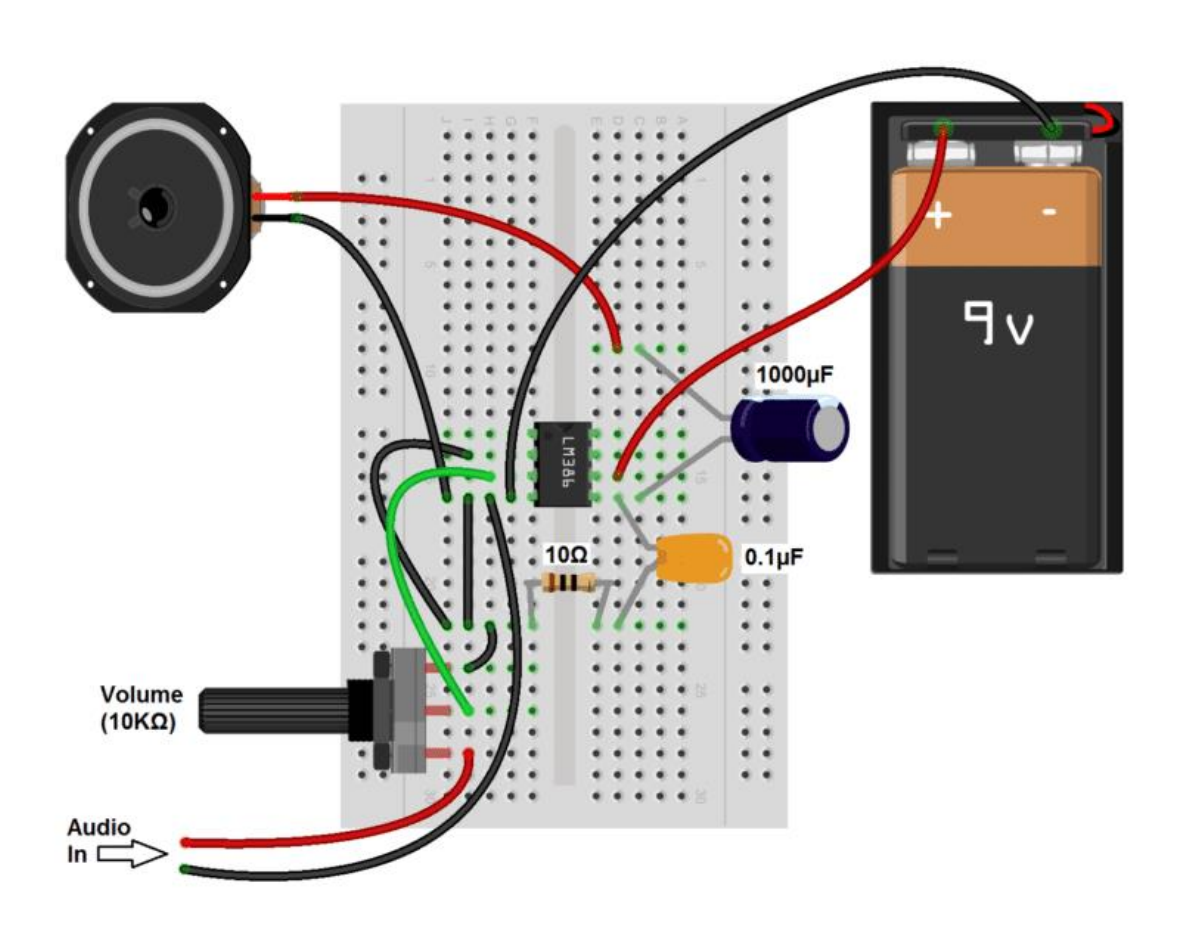


في مخطط الأسلاك أعلاه ، تتدفق ال Audio in اي الصوت المدخل الى أرض إدخال الصوت عبر نفس مسار أرض إخراج الصوت.

تكون أرض الإخراج "صاخبة" اي فيها ضجيج وستتسبب في تشويه إشارة الدخل إذا كانت سلكية بهذه الطريقة. وتعتبر الأرض المدخلة للصوت حساسة لأي تداخل وأي ضجيج يتم التقاطه من خلال مكبر الصوت.

لذلك اجعله هدفًا للاحتفاظ بأرض الإدخال منفصلة عن المسارات الأرضية الأخرى قدر الإمكان.

على سبيل المثال ، يمكنك توصيل تزويد الطاقة وإدخالها وإخراجها مباشرة إلى الدبوس الأرضي (رقم 4) في LM386 مثل هذا:

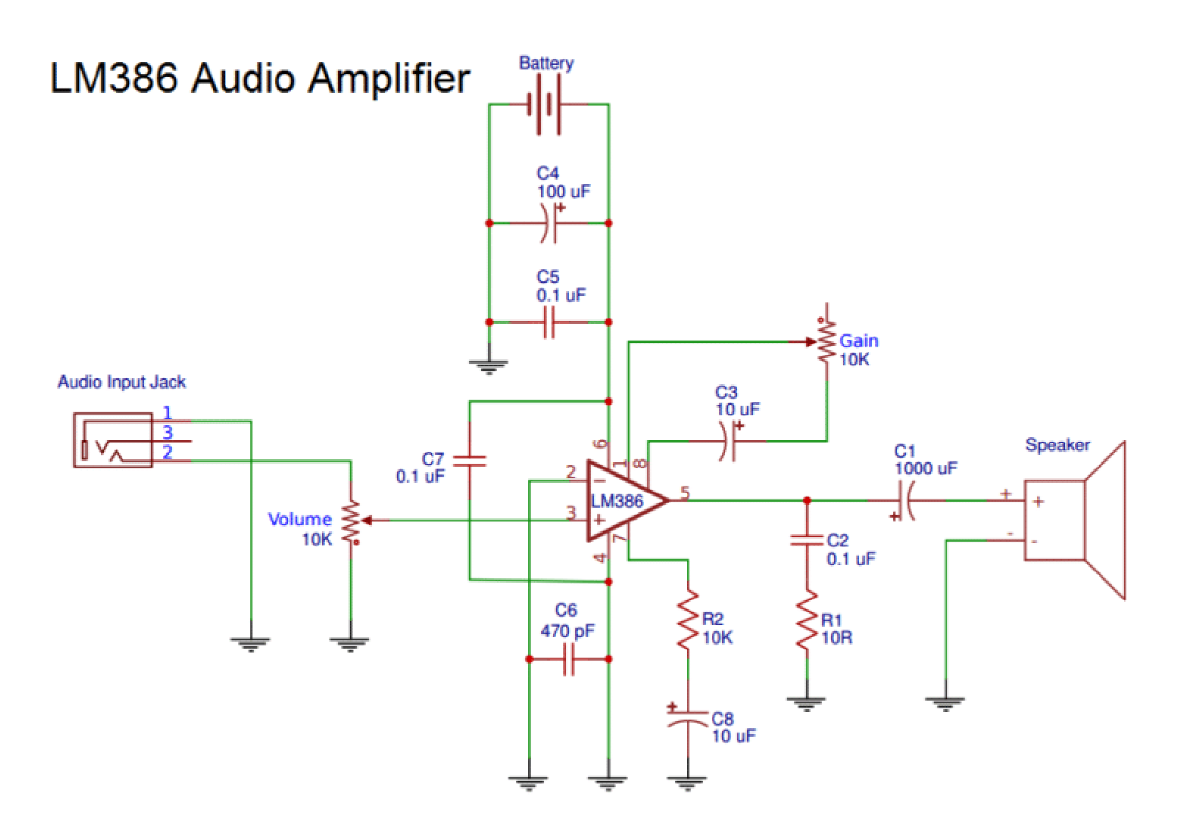


سيؤدي ذلك إلى تقليل المسافة التي تتدفق بها أرض الإدخال عبر أرض الإخراج.

يجب أن يكون توصيله هكذا أفضل من الدائرة الأولى ، لكنك ربما لا تزال تلاحظ بعض الضوضاء والثبات. سنقوم بإصلاح ذلك في الدائرة التالية عن طريق إضافة المكثفات فك الارتباط وعوامل تصفية RC .

الآن وبعد أن رأيت الحد الأدنى من ما يلزم لإنشاء مضخم صوت مع LM386 سننتج لك إنشاء إصدار أعلى دقة مع التحكم في الكسب القابل للتعديل.

ملاحظة: معظم قيم المكونات في هذه الدائرة ليست مهمة. إذا لم يكن لديك قيمة معينة فحاول استبدال شيء قريب وستعمل على الأرجح.




العديد من الأشياء في هذه الدائرة تجعل الأمر يبدو أفضل:

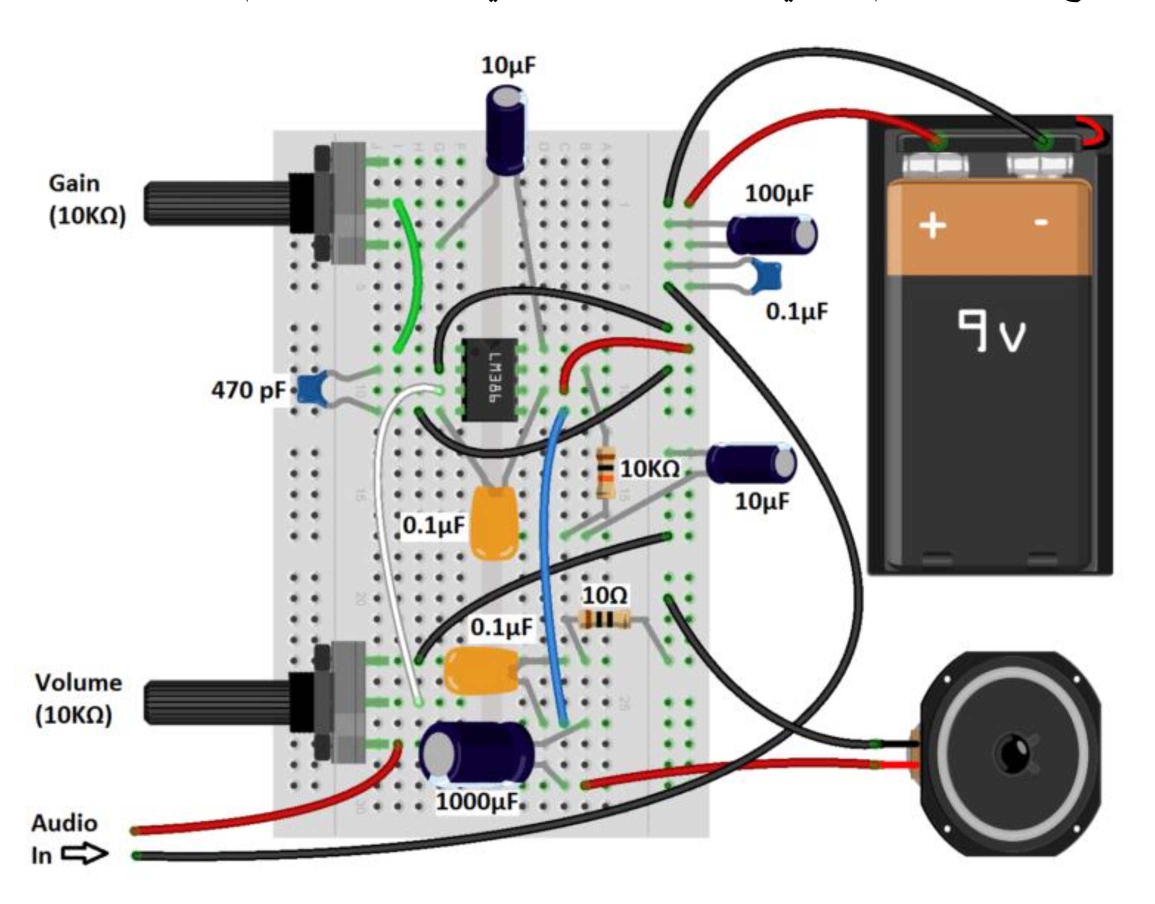
مكثف 470 pF بين إشارة الدخل الموجبة والأرض ، والذي يقوم بتصفية التداخل الذي تم التقاطه بواسطة أسلاك إدخال الصوت.

المكثفات 100 µF و 0.1 µF بين دبابيس الطاقة الموجبة والسالبة لفصل مزود الطاقة. سيقوم المكثف 100 µF بترشيح ضوضاء منخفضة التردد بينما يقوم مكثف 0.1 µF بترشيح ضوضاء عالية التردد.

مكثف 0.1 µF بين السنين 4 و 6 ، لفصل إضافي لمزود الطاقة إلى الشريحة.

مقاومة 10 كيلو أوم ومكثف 10 µF على التوالي بين دبوس 7 والأرض لفصل إشارة إدخال الصوت.

يوضح لك هذا الرسم البياني كيفية توصيل كل شيء إذا كنت تستخدم لوحة:



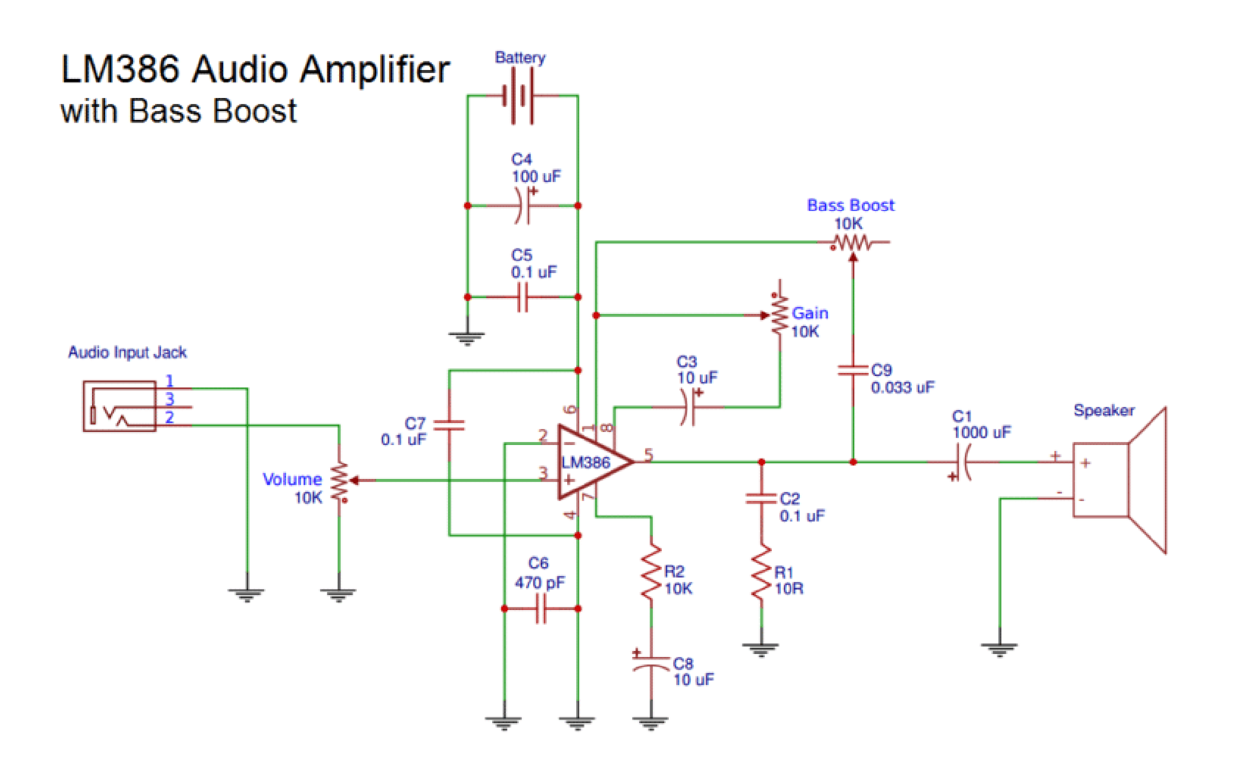
شيء مهم يجب مراعاته عند توصيل أي مضخم صوت هو أن أنقى صوت سينجم عن إبقاء جميع اتصالات الأسلاك والمكونات قريبة من الشريحة قدر الإمكان لذلك يجب الحفاظ على الأسلاك قصيرة قدر الإمكان هذا سوف يساعد لتقليل الضجيج .

## مكبر الصوت LM386 مع تعزيز BASS

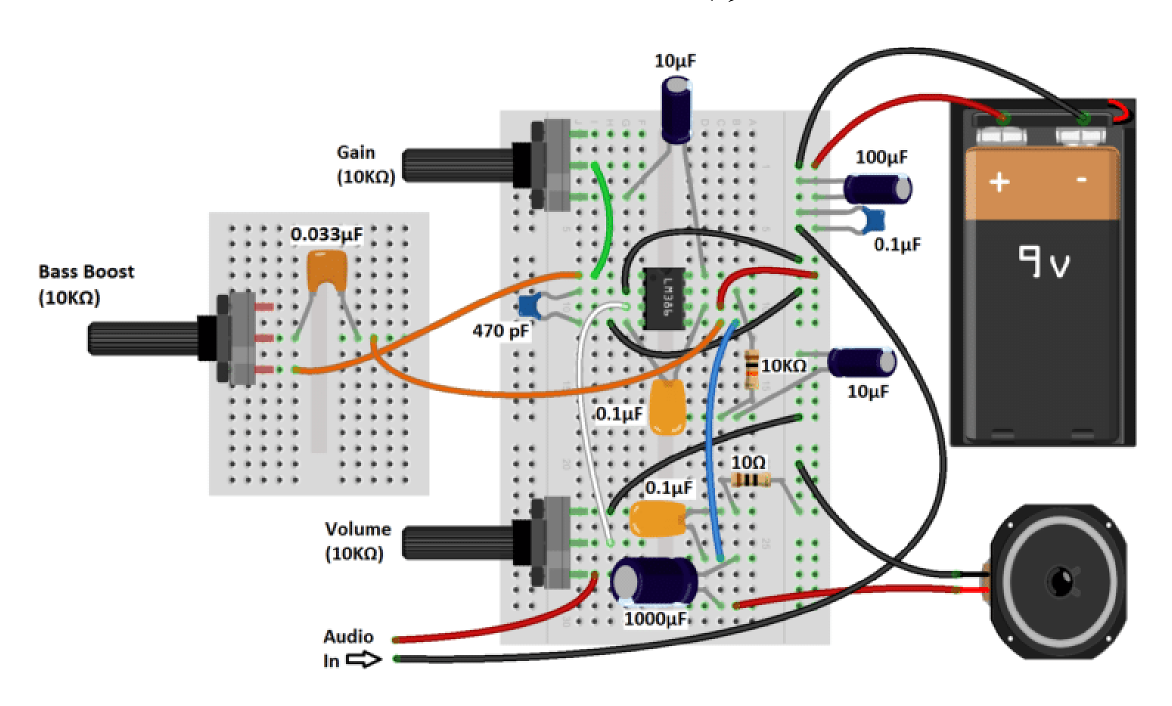
ميزة رائعة من LM386 هو خيار لإضافة دفعة باس قابل للتعديل إلى مكبر للصوت. ربما ستجد أن هذه هي أفضل دائرة تضخيم صوت.

يعد (bass boost) ببساطة مرشح تمرير منخفض ، ويزيل معظم الضوضاء التي لا تزيلها المكثفات الفاصلة.

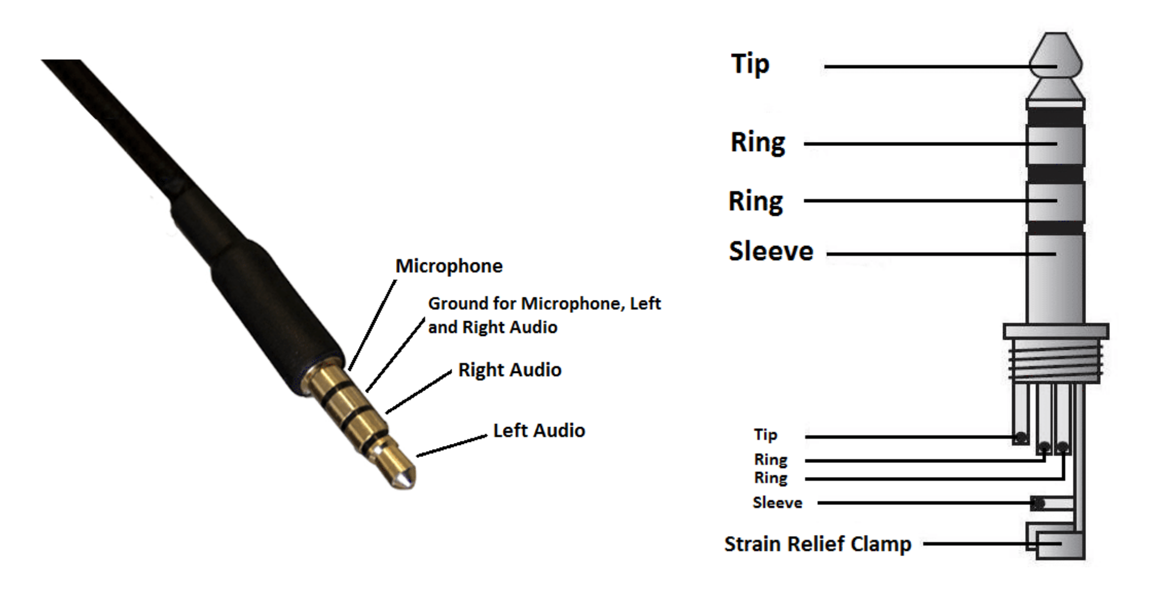
كل ما تحتاجه لبناء دائرة (BASS) هو مكثف 0.033 µF ومقاومة متغيرة 10كيلو أوم على التوالي بين الطرفين 1 و 5:

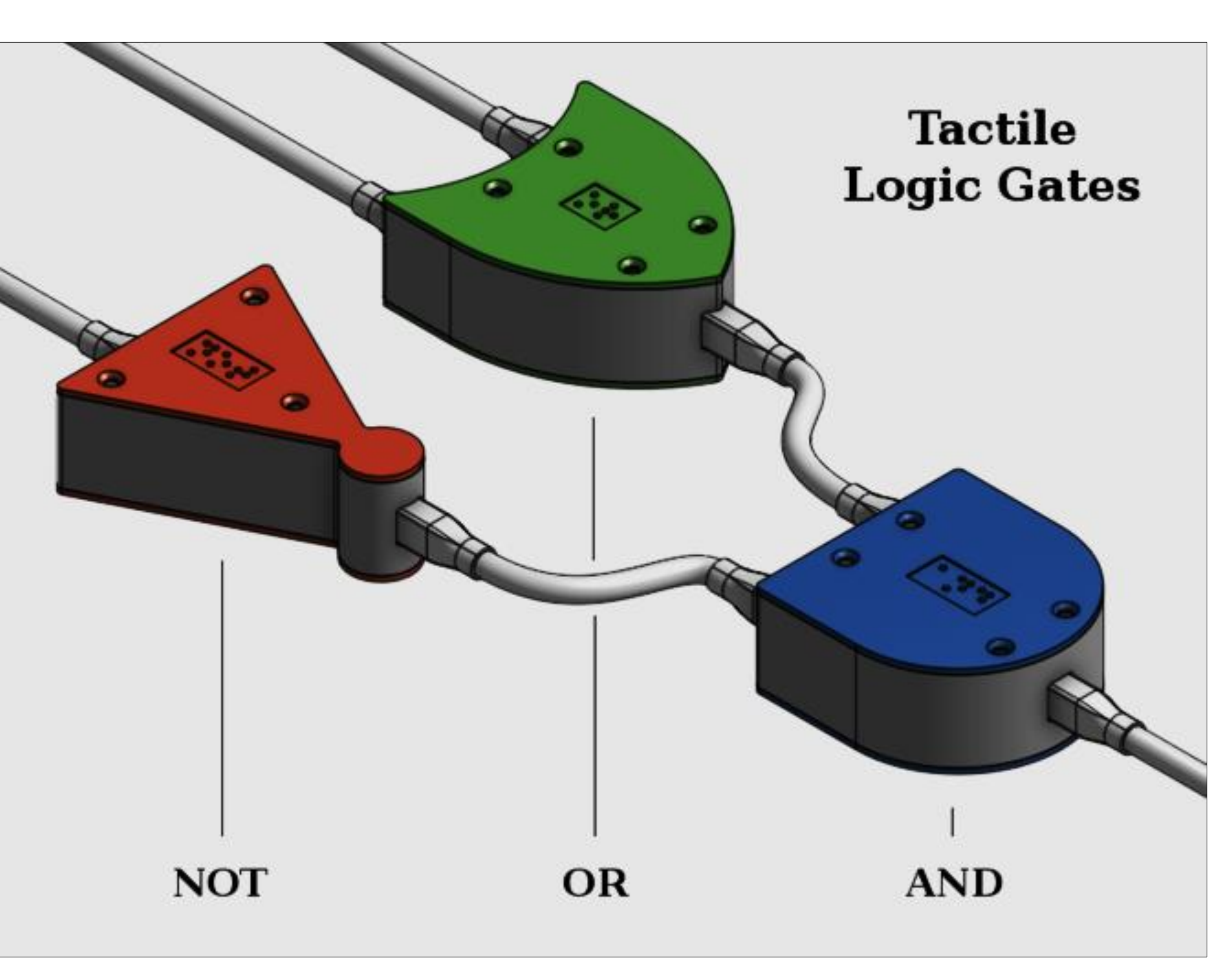


إليك مخطط الأسلاك:



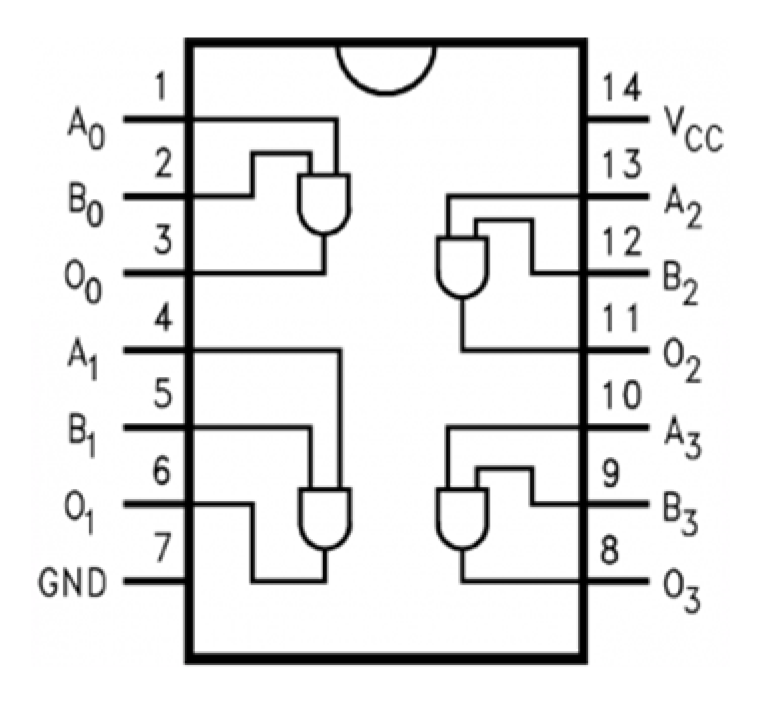

إحدى الطرق السهلة لتوصيل إدخال الصوت (Audio In) في هذه الدوائر هي قطع مقبس الصوت 3.5 مم من مجموعة قديمة من سماعات الرأس وأسلاكها إلى دبابيس اللوح.






البوابة المنطقية عبارة عن عنصر إلكتروني رقمي، يقوم بتنفيذ تابع منطقي معين .

البوابات المنطقية (وهي أحد مكونات بناء الكثير من الدوائر المتكاملة ذاتها) يمكن أن تأتي على شكل حزم دوائر متكاملة مستقلة خاصة بها. حيث يمكن أن تحتوي الدوائر المتكاملة الخاصة بالبوابات المنطقية على مجموعة من البوابات في حزمة واحدة، مثل بوابة AND رباعية الدخل التالية:




يمكن أن يتم توصيل البوابات المنطقية بداخل الدوائر المتكاملة لإنشاء المؤقتات والعدادات (counters) والمساكات (latches) ومسجلات الإزاحة والعديد من الدوائر المنطقية الأخرى. معظم هذه الدوائر البسيطة تأتي على شكل حزم DIP أو SOICأو.SSOP

## الجبر البولياني Boolean Algebra

الصناعة التقنية الحالية، والأجهزة الحاسوبية التي نستخدمها اليوم – على مختلف أنواعها – وكافة التقنيات المعقدة التي يستخدمها العلماء في أبحاثهم ودراساتهم، تعتمد جميعها على مبدأ بسيط واحد، وهو تمثيل المعلومات باستخدام عددين، هما الصفر (0) والواحد (1). هذين العددين، هما الأساس في كل ما يتعلق بالحواسيب، والدارات المتكاملة، وكل أشكال المعلومات، من صورٍ لفيديوهات لنصوص لبرامج وتطبيقات وغيرها، ليست سوى أشكال متنوعة لسلاسلٍ طويلة من الأصفار والواحدات.

الكلام السابق يتطلب التوضيح، وهو ما نود القيام به عبر هذا الدرس، الذي سنخصصه للحديث عن أسس المنطق الرقمي، وعلاقته بالإلكترونيات الرقمية، والدارات المتكاملة، والمعالجات الحاسوبية، وكافة المظاهر التقنية الحديثة التي نستخدمها بحياتنا اليومية، ربما بدون أن نعرف كيف تعمل بشكلٍ فعليّ.

جبر بول , قبل الحديث عن العناصر الإلكترونية الرقمية وكيفية عملها، وكيفية قيامها بمعالجة المعلومات والبيانات، علينا أن نسلط الضوء على أحد الفروع الهامة في الرياضيات، وهي "الجبر البولياني."Boolean Algebra

تاريخياً، فإن الجبر البولياني قد تم ابتكاره عبر عالم الرياضيات البريطاني "جورج بول George Boole"، وقد قدمه للمرة الأولى في كتابه "التحليل الرياضي للمنطق The Mathematical Analysis of Logic" والذي صدر عام 1847. في عام 1854، قام جورج بول بتقديم أسس الجبر البولياني بشكلٍ واسع في كتابه الأشهر "دراسة في قوانين التفكير An Investigation of the Laws of Thought".

ومن الاسم، فإن الجبر البولياني هو أحد فروع علم الجبر في الرياضيات، ولكنه بخلاف الجبر الاعتيادي، فإنه يفترض تواجد المتحولات الرياضية ضمن ما يعرف بـ "قيم الحقيقة "Truth Values وهي: القيمة الحقيقيةTrue ، القيمة الخاطئة .False بالتالي، فإن كل الأرقام والأعداد التي يتم التعامل معها في الجبر العادي، تتحول في الجبر البولياني لتركيباتٍ من الحالات الحقيقية Truth والخاطئة .False لسهولة التعامل، تم إسناد قيمة "1" للحالة الصحيحة، وقيمة "0" للحالة الخاطئة. وبالتالي، فإن أساس الجبر البولياني هو التعامل مع كافة المعطيات استناداً لقيمتين مرجعيتين: 0 و 1.

الاختلاف الآخر الذي يميز الجبر البولياني عن الجبر العادي هو العمليات الرياضية، ففي حين أن الجبر العادي يعتبر أن العمليات الأساسية فيه هي الجمع والطرح والضرب والقسمة، تعتبر العمليات الأساسية في الجبر البولياني هي: عملية الاقتران Conjunction، وعملية الفصل-اللاقترانDisjunction ، وعملية النفي .Negationإذاً، فإن أساس الجبر البولياني هو قيمتين تدعيان بقيم الحقيقة، وثلاثة عمليات رياضية أخرى.

لاحقاً، أصبحت القيم المنطقية التابعة للجبر البولياني تعرف باسم "الأرقام الثنائية Binary Digit" أو اختصاراً "بت -bit". كما أن العمليات الأساسية أصبحت تعرف بـ:

1 -عملية الاقتران :Conjunction أو عملية الضرب المنطقي AND

2 -عملية الفصل – اللااقتران :Disjunction أو عملية الجمع المنطقي OR

3- عملية النفي Negation: أي النفي المنطقي NOT

الآن لنفرض أنه يوجد لدينا متحولين، (x) و (y)، وليكن كل منهما مُتحوّل منطقي تابع لجبر بول، فإن كل من المتحولين يمكن أن يأخذ أحد قيمتين، الصفر أو الواحد. يتم تعريف عملية الضرب المنطقي AND بشكلٍ مشابه لعملية "التقاطع" الجبرية، أي أن ناتج عملية الضرب المنطقي هو كافة القيم المشتركة بين المتحولين. فإذا فرضنا أن المتحول (x) قيمته هي (0) والمتحول (y) قيمة هي (1)، فإن ناتج عملية الضرب المنطقي لهما ستكون (0)، لعدم وجود أي قيمة مشتركة بينهما، ويكون ناتج الضرب المنطقي لهما مساوياً للواحد في حالة واحدة فقط، وهي إذا كانت قيمتيهما معاً تساوي الواحد. بالنسبة لعملية الجمع المنطقي OR، فهي العملية التي تكافئ عملية "الاجتماع" الجبرية، أي أن ناتج عملية الجمع المنطقي هو كافة القيم المشتركة وغير المشتركة بين المتحولين، وبالتالي إذا كانت قيمة المتحول (x) هي (0) وقيمة المتحول (y) هي (1)، فإن ناتج عملية الجمع المنطقي بينهما ستكون (1)، وليس (0). أخيراً، فإن عملية النفي المنطقي تعني أمر واحد بسيط: إذا كانت قيمة المتحول (x) هي (0) فإن ناتج تطبيق عملية النفي المنطقي عليه ستكون (1)، أي أن عملية النفي المنطقي تؤدي للحصول على النتيجة المعاكسة لقيمة المتحول نفسه.

يمكن تلخيص نتائج الكلام السابق وفقاً للجدول التالي، الذي يبين العمليات المنطقية الأساسية ورموزها، ونتائج تطبيقها على المتغيرين (x) و (y)

العمليات المنطقية الأساسية في جبر بول ورموزها

| x | y | x^y (AND) | xᵛy (OR) | ¬x (NOT) | ¬y(NOT) |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 |

هنالك ملاحظة هامة تتعلق بالعمليات الأولية المتعلقة بالجبر البولياني، وهي تتعلق برموز العمليات، فمعظم الكتب والمراجع الأكاديمية لا تستخدم الرموز الموجودة هنا، بل تستخدم نفس رموز الجبر العادي، أي إشارة ( . ) للإشارة لعملية الضرب المنطقي، وإشارة ( + ) للإشارة لعملية الجمع المنطقي، ويتم استخدام خط صغيرة ( – ) يوضع فوق المتغير، للإشارة لعملية النفي. السبب هنا هو تبسيط إجراء العمليات، وربطها بالذهن أكثر مع عمليات الجبر العادي، إلا أنه يجدر أن نشير أن الرموز الواردة في الجدول هي الرموز الأساسية للعمليات المنطقية في الجبر البولياني، ولا يصح استخدام رموز الجبر العادي ضمن عمليات الجبر البولياني، من حيث المبدأ.

العمليات المشتقة

بناءً على العمليات الأساسية في الجبر البولياني، يمكن أن يتم اشتقاق عمليات أخرى، بحيث تستطيع أن تقوم بتنفيذ مهام أخرى على المعطيات والأرقام الثنائية (البتات). العمليات المشتقة الأساسية في الجبر البولياني (المنطقي) هي:

1 -عملية "التضمين."Implication

2 -عملية "الجمع الحصري "Exclusive OR والتي يشار لها غالباً بالاختصار "XOR".

3- عملية "التكافؤ المنطقي Exclusive NOR" والتي يشار لها غالباً بالاختصار ."XNOR"

الآن، وبالعودة للفرض الأساسي لدينا، وهو متحولين منطقيين (x) و(y) ، فإنه يمكننا تعريف العمليات المنطقية المشتقة كما يلي: بالنسبة لتطبيق عملية التضمين على متحولين منطقيين (x) و(y) ، فإن ناتج العملية سيأخذ أحد شكلين، فإذا كانت قيمة المتحول (x) هي الواحد، فإن ناتج العملية سيكون قيمة المتحول (y) مهما كانت، وإذا كانت قيمة المتحول (x) هي (0)، فإن ناتج العملية سيكون هو 1

بالنسبة لعملية"XOR" ، فإن ناتج تطبيق عملية XOR على المتحولين المنطقيين (x) و (y) سيكون "1" في حال كانا ذا قيمٍ مختلفة، وسيكون "0" في كان لهما نفس القيمة.

أخيراً، فإن عملية "XNOR" المنطقية تمثل نفي عملية"XOR" ، أي أنه يمكننا الحصول على نتيجة العملية عبر نفي نتائج عملية.XOR

الجدول التالي يوضح العمليات المشتقة، ونتائج تطبيقها على المتحولين المنطقيين (x) و (y):

## العمليات المنطقية المشتقة في جبر بول ورموزها

| x | y | $x \rightarrow y$ (implication) | $x \oplus y$ (XOR) | $x \equiv y$ (XNOR) |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |

بالنسبة للعمليات المشتقة، فإنها حتماً لم تأتي من فراغ، أو من مجرد تأويلاتٍ ممكنة للحالات المختلفة للقيم المنطقية، بل تم التوصل إليها عبر المعادلات الرياضية الخاصة بالجبر البولياني، والجدول التالي، يوضح العبارات الرياضية الخاصة بالعمليات المشتقة، وكيف يمكن صياغتها باستخدام المعاملات الجبرية المنطقية الأساسية، ومعاملات الجبر العادي، وذلك بهدف التبسيط والتقريب من الذهن. يجدر بنا التنويه إلى الجدول يتضمن المعادلات، بحيث تكتب المعادلة بالمرة الأولى وفقاً لصياغة الجبر البولياني، والمرة الثانية نفس المعادلة ولكن باستخدام المعاملات الجبرية العادية. هنالك أمر آخر هام، وهو أن نفي متحول منطقي واحد يتم باستخدام إشارة ( ' )، بينما نفي متحولين ضمن قوس، يتم بوضع سطر طويل فوق القوس كاملاً:

## مُعادلات العمليات المشتقة في جبر بول

عملية التضمين Implication

$$x \rightarrow y = \neg x \vee y$$

$$x \rightarrow y = \acute{x} + y$$

عملية الجمع الحصري XOR

$$x \oplus y = (x \vee y) \wedge \neg (x \wedge y)$$

$$x \oplus y = (\acute{x}.y) + (\acute{y}.x)$$

عملية التكافؤ XNOR

$$x \equiv y = \neg (x \oplus y)$$

$$\overline{(x \oplus y)} = (x.y) + \overline{(x.y)}$$

## قوانين الجبر البولياني

كما الجبر العادي، فإن الجبر البولياني يتميز أيضاً بالعديد من القوانين التي توضح العمليات بين المتحولات المنطقية البوليانية. مثل الجبر العادي، هنالك قوانين توزيعية وتجميعية للعمليات الأساسية الخاصة بالجبر البولياني، ولن نتطرق لها لأنه يمكن القول عنها أنها عمليات بديهية. ما يهمنا هنا، هو القوانين الخاصة والتي لن يتم مشاهدتها سوى في الجبر البولياني.

ومن أجل أفضل إيضاح ممكن، سنقوم بعرض هذه القوانين في الصورة التوضيحية التالية، والتي سنكتب فيها أيضاً القوانين ضمن صيغتين: الصيغة الأولى بحسب الشكل الأصلي للمعاملات الجبرية في جبر بول، والثانية بحسب المعاملات الجبرية العادية.

### قوانين الجبر البولياني

**قوانين التكميل**

a. $x \wedge (\neg x) = 0$ or $x . x' = 0$

b. $x \vee (\neg x) = 1$ or $x + x' = 1$

**قانون النفي المضاعف**

$\neg(\neg x) = x$ or $(x')' = x$

**قوانين دي مورغان**

a. $\neg x \wedge \neg y = \neg(x \vee y)$ or $x' . y' = \overline{(x + y)}$

b. $\neg x \vee \neg y = \neg(x \wedge y)$ or $x' + y' = \overline{(x . y)}$



بالنسبة لقوانين التكميل، فالقانون الأول يعني أن حاصل عملية الضرب المنطقي لمتحول منطقي (x) مع النفي الخاص به سيساوي الصفر، ولفهم القانون أكثر، علينا أن نتذكر أن نقوم بتشبيهه بعملية التقاطع: إذا كان لدينا متحول منطقي (x) قيمته هي "1"، وبالتالي فإن نفي هذا العنصر سيكون ذو قيمةٍ تساوي "0"، وبالتالي لن يتواجد أي شيء مشترك بينهما، أي أن تقاطع هذين المتحولين سيكون مساوياً للصفر.

بالنسبة لقانون التكميل الثاني، فهو يعني أن حاصل عملية الجمع المنطقي لمتحول منطقي (x) مع النفي الخاص به سيساوي الواحد. وبشكلٍ مشابه للقانون الأول، سنتذكر أن عملية الجمع المنطقي تشابه عملية الاجتماع الجبرية، وهي تعني كافة الحدود المشتركة وغير المشتركة بين المتحولين، وبالتالي إذا كان لدينا متحول منطقي (x) قيمته هي "0"، فإن نفيه سيكون ذو قيمة تساوي "1"، وحاصل اجتماعهما سيكون مساوياً للواحد. يمكن الحصول على نفس النتيجة إذا كانت قيمة المتحول (x) هي الواحد.

بالنسبة لقانون النفي المضاعف، فهو سهل جداً من حيث الفهم. لنفرض أن لدينا متحول منطقي (x) قيمته هي الصفر، إذا قمنا بنفيه للمرة الأولى، ستكون النتيجة هي الواحد، وبإجراء نفي ثاني للنتيجة، ستكون النتيجة هي الصفر. بالمختصر، يمكن الحصول على قيمة المتحول نفسها عبر نفيه مرتين.

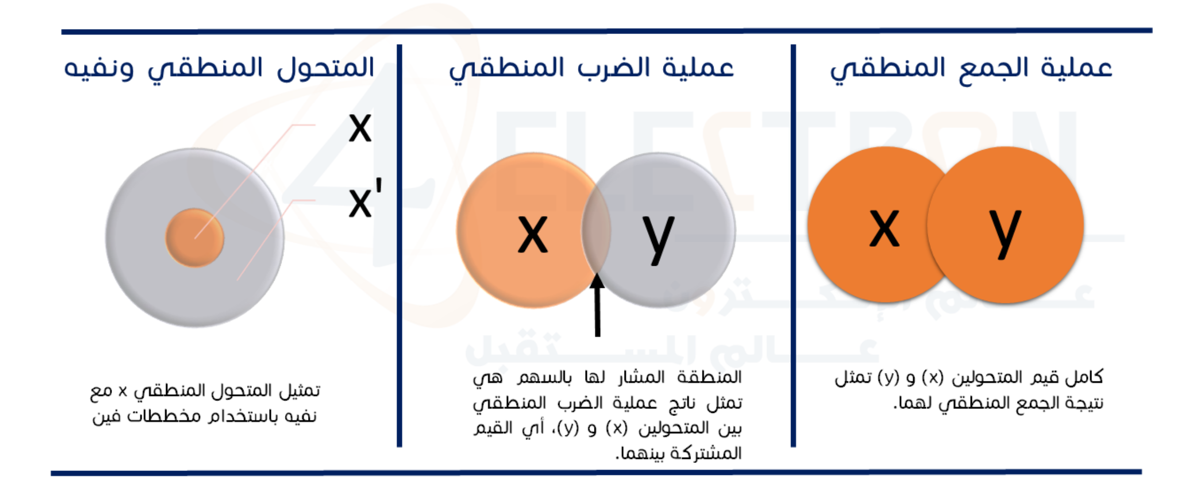
قوانين دي مورغانDe Morgan Laws ، هي من أهم قوانين الجبر البولياني، وهي تساهم بشكلٍ كبير بمعرفة ناتج أو خرج البوابات المنطقية الإلكترونية. قانون دي مورغان الأول يعني ما يلي: حاصل عملية الضرب المنطقي لنفي متحولين منطقيين (x)و(y) ، سيكون عبارة عن نفي نتيجة الجمع المنطقي للمتحولين. لنفهم القانون أكثر، لنفرض أن قيمة المتحولين هي "0"، وبالتالي فإن قيمة نفي المتحولين ستكون "1"، وأخيراً، فإن ناتج عملية الضرب المنطقي لقيمتين كل منهما "1" فهي "1". لنأخذ الشطر الثاني من العبارة، ولنتذكر أن قيمة المتحولين هي "0"، وبالتالي فإن ناتج جمعهما منطقياً هي "0"، الآن بأخذ نفي النتيجة، نحصل على "1"، وبالتالي العبارة صحيحة بسبب تساوي طرفيها. يمكن تطبيق الطريقة السابقة عبر إسناد قيمٍ مختلفة للمتحولين (x)و (y) وملاحظة خرج كل طرف من طرفي العبارة. كما يمكنٍ أيضاً البرهان على صحة قانون دي مورغان الثاني بنفس الطريقة أيضاً، أي عبر إسناد قيمٍ مختلفة للمتحولات المنطقية، ومن ثم الحصول على خرج كل طرف من طرفي العبارةٍ، والتي يجب أن تكون متساوية تماماً.

تمثيل عبارات وعمليات جبر بول باستخدام مخططات فين

تعتبر مخططات فين Venn Diagrams من أهم الطرق البيانية التي تستخدم لتوضيح العمليات الجبرية، مثل التقاطع والاجتماع والنفي. وبما أن عبارات الجبر البولياني تتشابه مع هذه المفاهيم، فإنه يمكن أيضاً استخدام مخططات فين من أجل توضيح العبارات والعمليات المنطقية المختلفة.

سنوضح كيف يمكن استخدام مخططات فين من أجل إظهار (3) عمليات منطقية أساسية: عملية الضرب المنطقي، عملية الجمع المنطقي، وعملية النفي المنطقي، وذلك من أجل متحولين منطقيين (x) و (y).

## تمثيل عمليات جبر بول باستخدام المخططات




التوابع الرياضية في الجبر البولياني

مثل الجبر العادي، فإنه يمكننا كتابة توابع رياضية في الجبر البولياني، وهي تدعى "التوابع المنطقية Logic Functions". التابع المنطقي هو عبارة رياضية، تصف العلاقة بين عدة متحولات منطقية، بدءاً من العمليات المنطقية الأساسية (الضرب المنطقي، الجمع المنطقي، النفي المنطقي) وصولاً للعمليات المشتقة، وذلك مهما كان عدد المتحولات المنطقية.

بشكلٍ بسيط، يمكن القول أن كافة العمليات الأساسية والمشتقة في الجبر البولياني تمثل توابع منطقية، فهي عبارات تصف العلاقة بين المتحولات المنطقية، ومن خلالها نستطيع معرفة نتيجة ضرب أو جمع متحولين منطقيين، أو أي عملية أخرى.

يرتبط بمفهوم التوابع المنطقية "جدول الحقيقة Truth Table". جدول الحقيقة هو الجدول الممثل للقيم الممكنة للمتحولات المنطقية المختلفة، وكذلك يظهر لنا نتيجة تطبيق التابع المنطقي الذي يصف العلاقة فيما بينها. لو عدنا للشكل (1) والشكل (2)، فإنهما فعلياً يمثلان جداول حقيقة للمتحولات المنطقية (x) و (y)، حيث يظهرا كل القيم الممكنة لهما، ويظهر أيضاً ناتج تطبيق عمليات الضرب المنطقي، والنفي المنطقي، والجمع المنطقي، وناتج تطبيق العمليات المشتقة، مثل XOR و XNOR على المتحولات المنطقية.

اذا جبر بول، هو مجرد طريقةٌ أخرى للنظر إلى المعطيات والبيانات، وقد ساهم عبر أسسه ومبادئه وعملياته بتقديم الأساس الصلب واللازم للثورة الرقمية، التي تم عبرها ابتكار الحاسوب الرقمي في خمسينيات القرن الماضي، وذلك بعد أن قام العالم الشهير كلود شانون بوضع أسس نظرية المعلومات، ومن ثم ومع انطلاق ثورة أنصاف النواقل على يد وليام شوكلي الذي ابتكر الترانزيستور، أصبح بالإمكان أن يصبح "الحاسوب" حقيقةً واقعية. المعلومات في الحاسوب ترمز اعتماداً على القيمتين المنطقيتين "0" و "1"، وتقوم عبارات الجبر البولياني والتوابع المنطقية بتوصيف عمل البوابات المنطقية الإلكترونية التي تشكل الدارات الحاسوبية، وبالتالي، فإنه لم يكن من الممكن الحصول على ثورتنا الرقمية اليوم، لولا الجبر البولياني وعملياته ومعادلاته.

# التابع المنطقي Logic Function

كما وضحنا في مقالاتٍ سابقة، فإن الجبر البولياني (أو الجبر المنطقي) هو فرعٌ خاص من فروع علم الجبر الرياضي، والذي يتميز بمتحولاته وأعداده الخاصة به. ولو عدنا للرياضيات التي نعرفها، فإن التابع الرياضي هو عبارة عن تطبيق، بحيث يكون دخله (X) وخرجه (Y). بنية وشكل التابع الرياضي هي ما سيحدد كيف سيكون الخرج (Y). فلو أخذنا كمثال بسيط التابع الرياضي التالي:

$$Y = F(x) = x^2$$

نحن ندعو (Y) الخرج، وندعو كل قيم (x) الممكنة بالدخل، أما شكل الخرج فهو يتحدد بشكل التابع، والذي هو $x^2$

الآن، وبحالة المتحولات المنطقية والجبر البولياني، فإن مفهوم التوابع موجود بنفس الطريقة وبنفس الأسلوب تماماً. فالتابع المنطقي عبارة عن علاقة بين مجموعة قيم تمثل الدخل، من أجل الحصول على الخرج. الفرق الأساسي بين التابع المنطقي والتابع الرياضي التقليدي، هو أن كافة قيم دخل وخرج التابع المنطقي ستكون قيم منطقية، أي أصفار وواحدات.

بشكلٍ أساسي، يوجد مجموعة من التوابع المنطقية والتي تمثل العمليات المنطقية الأساسية:

- عملية النفي المنطقيNOT
- عملية الضرب المنطقيAND
- عملية الجمع المنطقيOR
- عملية نفي الضربNAND
- عملية نفي الجمعNOR
- عمليةXOR
- عمليةXNOR

# جدول الحقيقة Truth Table

جدول الحقيقة هو عبارة عن ترتيب قيم الدخل الممكنة للتابع المنطقي مع قيم الخرج الممكنة له. فلو أخذنا أبسط تابع منطقي ممكن، وهو تابع عملية النفي، فإنه يمكننا توصيف خرج التابع بأنه معكوس أي دخل. فإذا كان الدخل هو "1" فإن الخرج سيكون "0"، وإذا كان الدخل هو "0" فإن الخرج سيكون "1". يمكن كتابة هذا الوصف عبر جدول الحقيقة التالي:

| الدخل | الخرج |
|---|---|
| 0 | 1 |
| 1 | 0 |

لو أخذنا تابعاً منطقياً له دخلين (على الأقل) مثل تابع الضرب المنطقي، فإننا سنقوم بما يلي:

سنسمي الدخل الأول (x) والدخل الثاني (y) والخرج هو نتيجة الضرب المنطقي لـ x و y. بما أننا نتملك دخلين، فإن عدد حالات الخرج الممكنة هو $2^2$ أي 4 قيم ممكنة للخرج. ترتيب هذا التوصيف ضمن جدول الحقيقة سيكون كما يلي:

| x | y | F = x.y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

الجدول اعلاه يمثل جدول الحقيقة لتابع AND المنطقي. إذاً، ومن أجل كتابة جدول الحقيقة الخاص بأي تابع منطقي (سواء كان من التوابع الأساسية أو كان تابعاً مركباً) فإن ما يلزمنا معرفته هو:

1. عدد متحولات الدخل المنطقية
2. معادلة التابع المنطقي

من المهم أن نعلم أن التوابع المنطقية ليست دوماً توابع بسيطة، والتوابع المنطقية الأساسية التي استعرضناها سابقاً هي أساس العمليات المنطقية، حيث يمكن كتابة معادلة تابع منطقي تشتمل على عدة عمليات منطقية متنوعة بنفس الوقت. بهذه الحالة سيكون جدول الحقيقة أكبر. بأي حال، فإننا يجب أن نتذكر على الدوام أي خرج أي تركيبة منطقية سيكون إما "0" أو "1."

البوابات المنطقية

الآن أصبح بإمكاننا الحديث عن البوابات المنطقية. وبالعودة للتعريف الذي بدأنا به، فإن البوابة المنطقية عنصر يقوم بتنفيذ تابع منطقي. هذا يعني أنه بالنسبة للعمليات المنطقية الأساسية والتوابع المنطقية الممثلة لها، يوجد عناصر أساسية تمثلها، وهي البوابات المنطقية.

تقسم البوابات المنطقية إلى: البوابات المنطقية الأساسية وهي تضم بوابات: NOT, AND, OR وإلى بوابات المستوى الثاني، وهي بوابات NAND, NOR, XOR, XNOR. وهي موضحة بمعرض الصور التالي

عند الحديث عن أي بوابة منطقية، يجب أن نتحدث عن الأمور التالية:

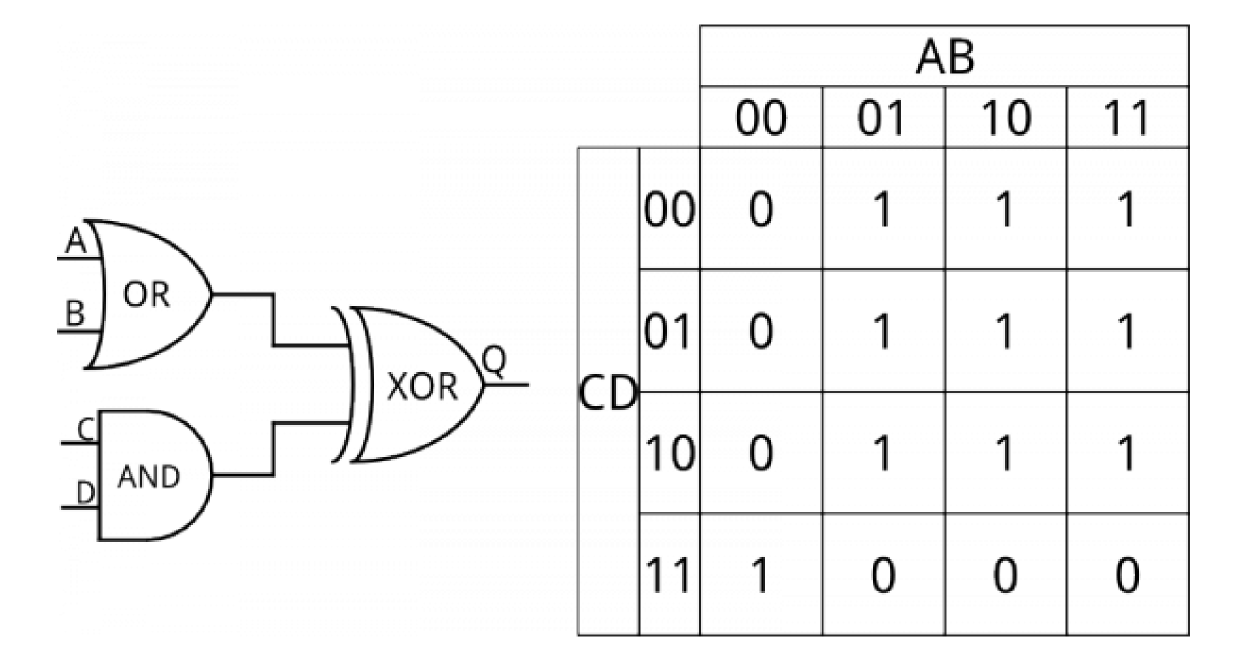
- رمز البوابة المنطقية
- التابع المنطقي الخاص بالبوابة المنطقية
- جدول الحقيقة الخاص بالبوابة المنطقية
- بنية البوابة المنطقية

وبما أننا أصبحنا نتملك فكرة عن المحددات الأساسية للبوابات المنطقية، فإننا سنقوم الآن باستعرض البوابات كاملةً مع محدداتها ضمن الجدول التالي:

<table>
<tr><th>جدول الحقيقة</th><th>التابع المنطقي</th><th>الرمز المنطقي</th><th>اسم البوابة</th></tr>
<tr><td><table><tr><th>x</th><th>y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></td><td>$F = x'$</td><td></td><td>بوابة النفي (العاكس)<br>NOT</td></tr>
<tr><td><table><tr><th>x</th><th>y</th><th>x+y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td>$F = x+y$</td><td></td><td>بوابة الجمع المنطقي<br>OR</td></tr>
<tr><td><table><tr><th>x</th><th>y</th><th>x.y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td>$F = x.y$</td><td></td><td>بوابة الضرب المنطقي</td></tr>
<tr><td><table><tr><th>x</th><th>y</th><th>(x+y)'</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td><td>$F = (x+y)'$</td><td></td><td>بوابة نفي الجمع<br>NOR</td></tr>
<tr><td><table><tr><th>x</th><th>y</th><th>(x.y)'</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td><td>$F = (x.y)'$</td><td></td><td>بوابة نفي الضرب<br>NAND</td></tr>
<tr><td><table><tr><th>x</th><th>y</th><th>x⊕y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td><td>$F = x \oplus y$<br>$F = (x.y') + (x'.y)$</td><td></td><td>بوابة الجمع الحصري<br>XOR</td></tr>
<tr><td><table><tr><th>x</th><th>y</th><th>(x⊕y)'</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td><td>$F = (x \oplus y)'$<br>$F = (x.y) + (x'.y')$</td><td></td><td>بوابة نفي الجمع<br>الحصري<br>XNOR</td></tr>
</table>

الجدول اعلاه كامل للبوابات المنطقية: يظهر الجدول رمز كل بوابة، مع التابع المنطقي الخاص بها، وجدول الحقيقة الذي يصف عملها.

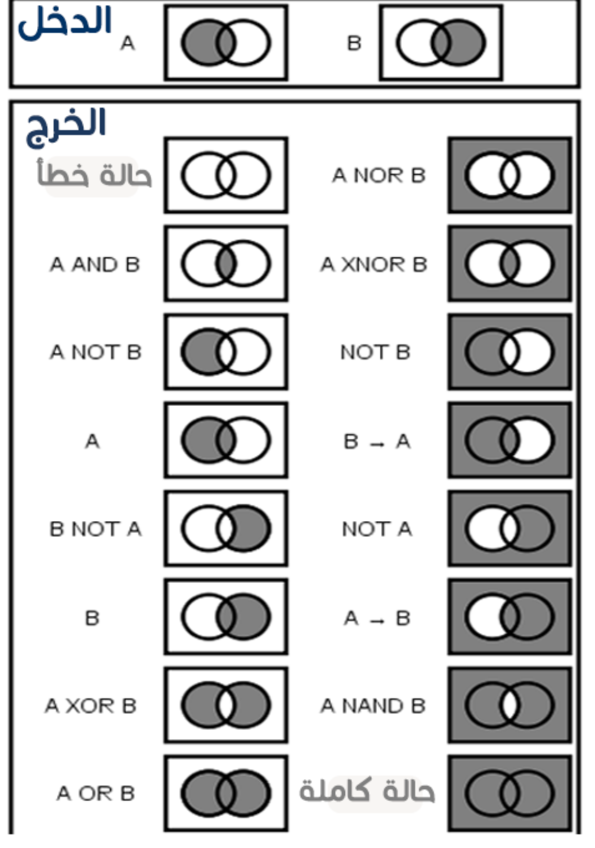
يمكن تمديد جداول الحقيقة إلى ما لا نهاية، إلى أي عدد من المُدخلات والمخرجات يُمكنك استيعابه قبل أن تفقد عقلك**!** الشكل التالي يوضح ما يبدو عليه جدول الحقيقة المُستخدم مع الدوائر ذات الأربعة أطراف دخل:




| | | AB | | | |
|---|---|---|---|---|---|
| | | 00 | 01 | 10 | 11 |
| CD | 00 | 0 | 1 | 1 | 1 |
| | 01 | 0 | 1 | 1 | 1 |
| | 10 | 0 | 1 | 1 | 1 |
| | 11 | 1 | 0 | 0 | 0 |

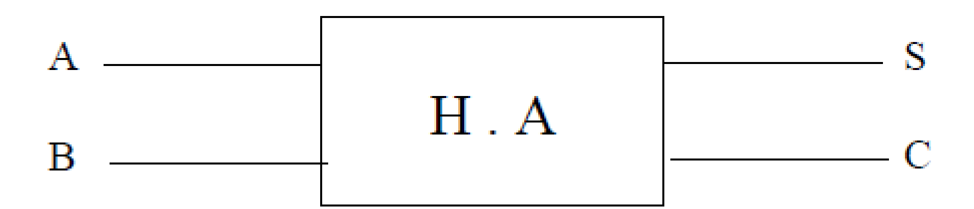
تعتبر مخططات فين أحد أفضل الأمثلة التوضيحية التي يمكن عبرها تمثيل العمليات الرياضية. وبحالة البوابات المنطقية، فإنه يمكن أن يتم استخدام مخططات فين من أجل تمثيل عمل كل بوابة.

سنقوم بتشكيل مخطط أول له رمز A ومخطط ثاني له رمز B. يمثل هذين المخططين دخل كل بوابة منطقية. القسم الأول من الجدول التالي يظهر الدخل بشكلٍ منفصل، ومن ثم الخرج. كل مرحلة من مراحل الخرج تمثل بوابةً منطقية. اللون الأبيض يمثل حالة "الخطأ المنطقي" أو العدد "0"، أما اللون الرمادي فيمثل حالة "حقيقة" منطقية أو العدد "1". تقاطع الألوان بين المخططين A وB يمثل خرج البوابة المنطقية.



# الجامع النصفي Half Adder H.A

الرمز




من الرمز يتضح ان الجامع النصفي له مدخلات ومخرجات ويتم الجمع بين رقمين هما A و B وناتج الجمع يكون على الخرج S من كلمة SUM اي المجموع والباقي على الخرج C من كلمة Carry وبالتالي يكون جدول الحقيقة له كالتالي :

| المدخلات | | المخرجات | |
|---|---|---|---|
| A | B | المجموع S | المرحل C |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

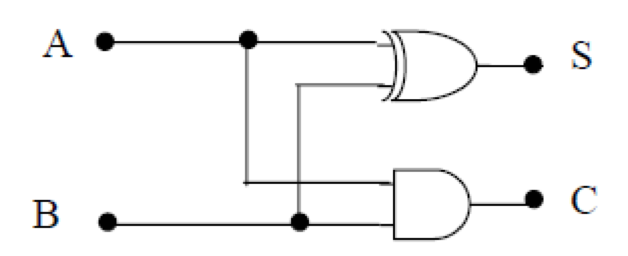
من جدول الحقيقة تكون معادلة المجموع :

$$S = \overline{A}B + A\overline{B}$$
$$S = A \oplus B$$

ومعادلة المرحل Carry :

$$C = AB$$

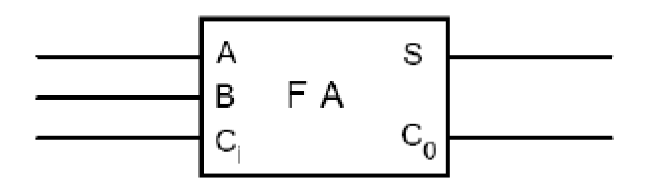
وهذه المعادلة تحققها بوابة "و"  And وبذلك يمكن تحقيق دائرة الجامع النصفي باستخدام بوابة  XOR و  AND كما في الشكل التالي :




## الجامع الكلي  Full Adder

تتقبل دائرة الجامع الكلي ثلاث مداخل وتعطي مخرجين هما المجموع والمرحل ان الفرق الاساسي بين  دائرة الجامع النصفي ودائرة الجامع الكلي , هو ان دائرة الجامع الكلي لها مدخل اضافي في المرحل السابق $C_i$

الرمز المنطقي لها :




جدول الحقيقة :

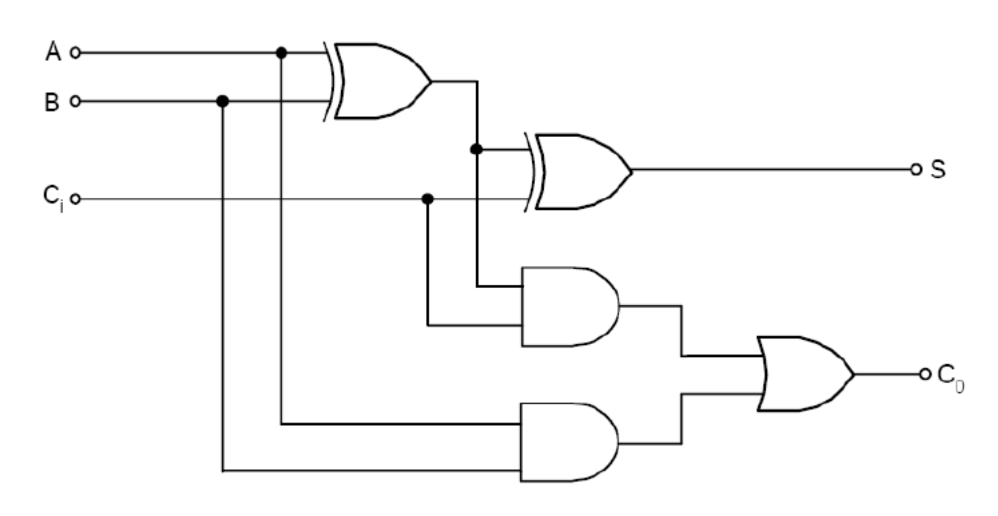
| A | B | $C_i$ | $C_0$ | S |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

يمكن استنتاج المعادلات المنطقية لمخرج الجامع الكلي كما يلي :

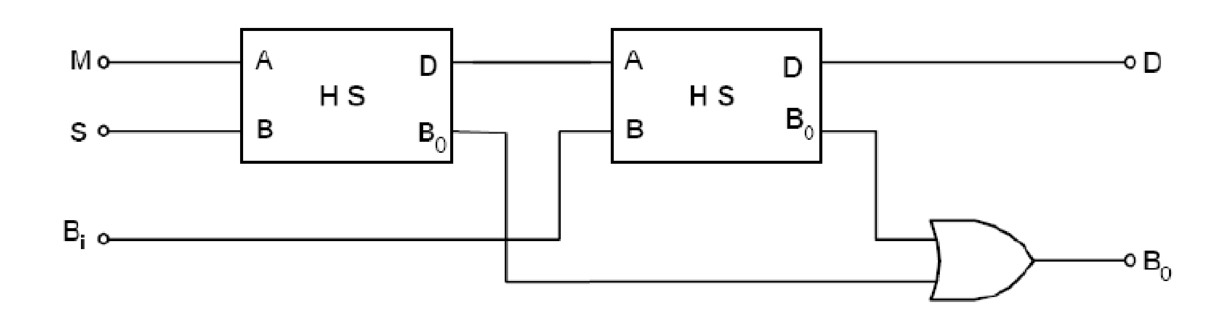
$$S = A \oplus B \oplus C_i$$

$$C_o = AB + (A + B)C_i$$

ويكون مخطط الدائرة المنطقية كالتالي :

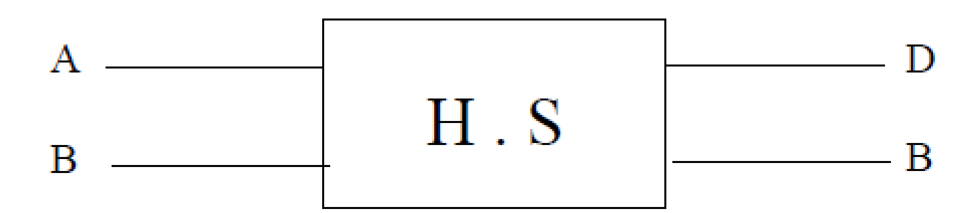



ويمكن تنفيذ الجامع الكلي باستخدام دائرة الجامع النصفي وبوابة OR :



## الطارح النصفي Half Subtract H.A

الرمز




من الرمز فان الطارح النصفي يجري الطرح بين خانتين ثنائيتين وتسمى عملية الطرح الناتجة في الخانة D من كلمة Difference "الفرق" اما عملية الاستعارة فتوضح قيمة العدد الناتج من الخانة B من كلمة Borrow "استعارة" ولقد صممت دائرة الطارح النصفي من خلال جدول الحقيقة لعمليات الطرح الثنائية لعددين فقط كالتالي :

| المدخلات | | المخرجات | |
|---|---|---|---|
| المطروح منه A | المطروح B | الفرق D | الاستعارة B |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

ومعادلة الفرق D :

يمثل بالبوابة XOR

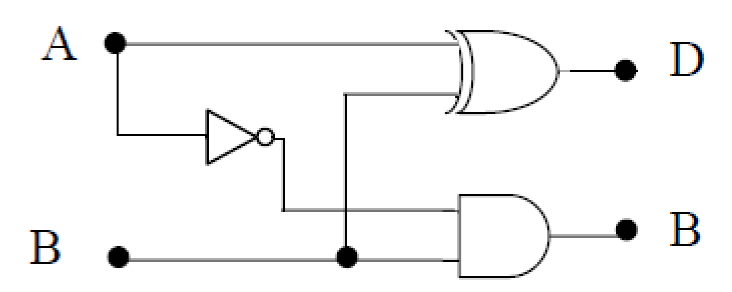
$$D= A\overline{B} + \overline{A}B = A \oplus B$$

ومعادلة B :

يمثل بالبوابة AND مع عاكس

$$B = \overline{A} \cdot B$$

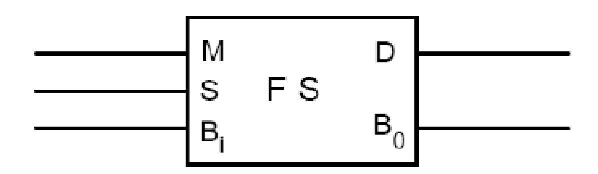
وتكون الدائرة المنطقية للطارح النصفي كما يلي :




## الطارح الكلي Full Subtractor

تستقبل دائرة الطارح الكلي ثلاثة مداخل وتولد خرج الفرق وخرج الاستعارة كما هو موضح في الشكل التالي

الرمز المنطقي :




جدول الحقيقة :

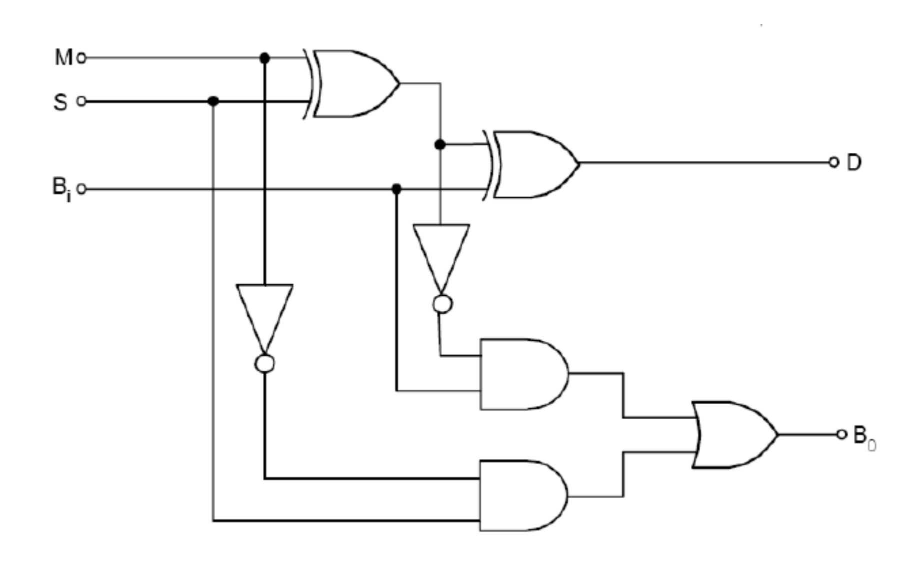
| M | S | $B_i$ | D | $B_0$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

ويمكن استنتاج المعادلات المنطقية لخرج الطارح الكي كما يلي :

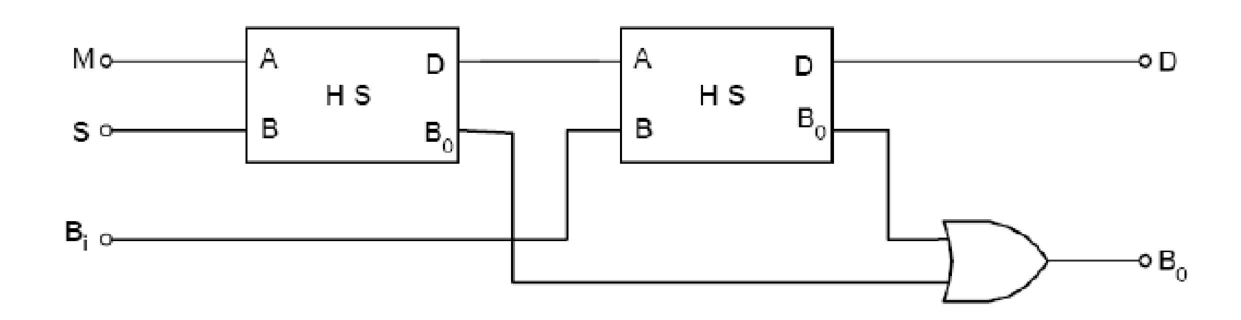
$$D = M \oplus S \oplus B_i$$

$$B_o = B_i + (M \oplus \overline{S})MS$$

ويتم تنفيذ الدائرة باستخدام البوابات المنطقية كما يلي :



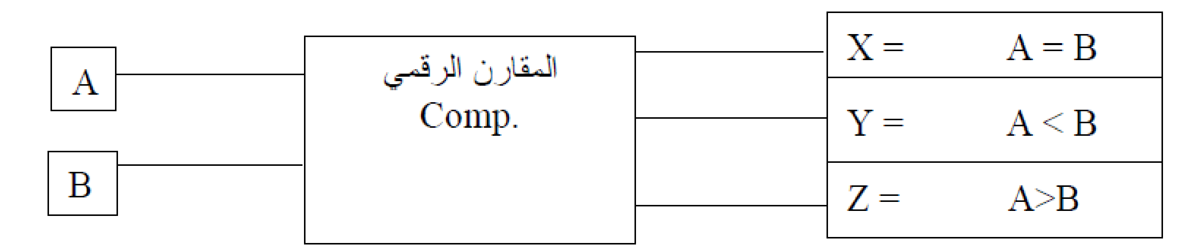

اما باستخام دائرة الطارح النصفي تكون النتيجة :



# Digital Comparator المقارن الرقمي

هو احد الدوائر التوافقية التي تقوم بالمقارنة بين عددين ثنائيين من حيث حالة اكبر من او اصغر من او حالة المساواة للعددين ( A>B, A<B , A=B)

ويكون رمز المقارن الرقمي كالتالي :




جدول الحقيقة للمقارن الرقمي :

| **A** | **B** | **X**<br>**A=B** | **Y**<br>**A<B** | **Z**<br>**A>B** |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 |

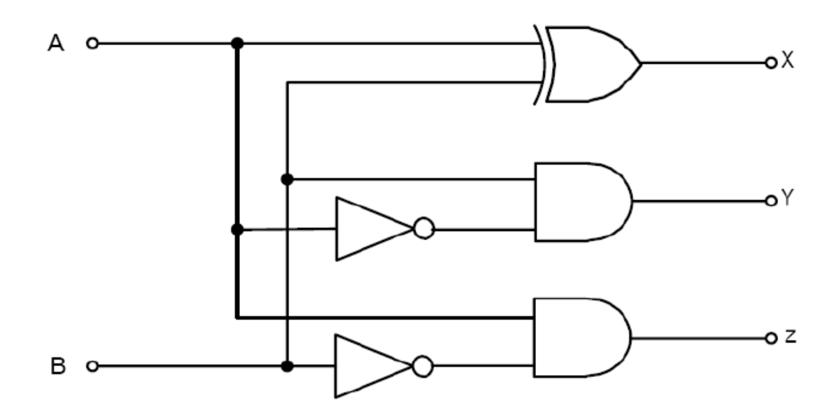
ومن الجدول نستنتج المعادلات التالية :

$$X = \bar{A}\bar{B} + AB = \overline{A \oplus B}$$

$$Y = \bar{A}B$$

$$Z = A\bar{B}$$

ومن المعادلات السابقة يمكن تمثيل المقارن الرقمي بالدائرة التالية :

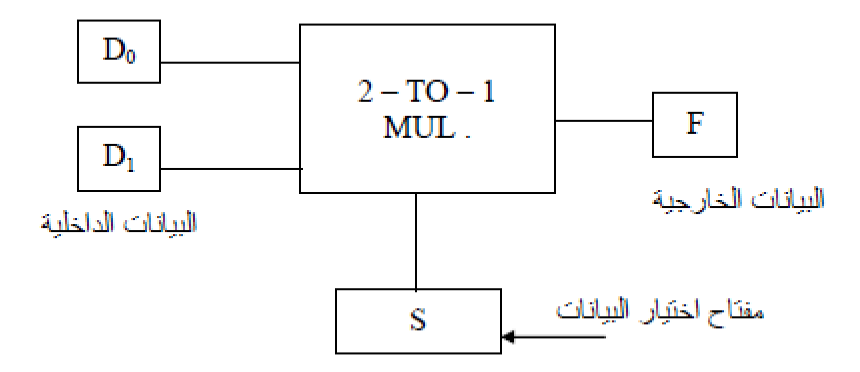


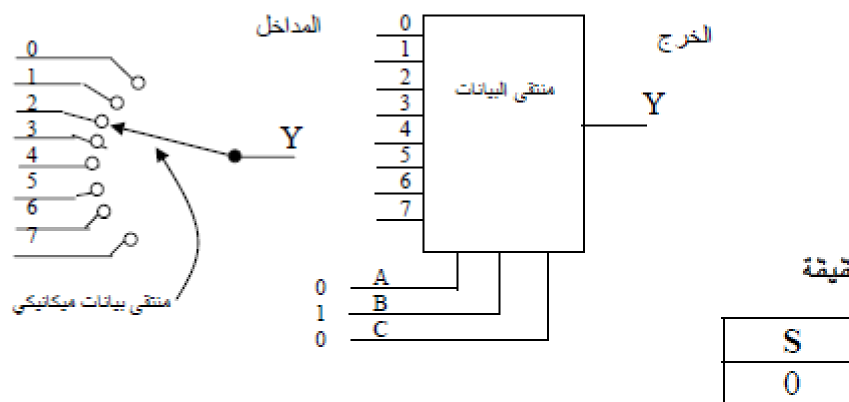

## منتقي البيانات Multiplexes

هو احد الدوائر المنطقية التوافقية ويكون في شكل دائرة متكاملة IC ويتكون من عدة بوابات منطقية (AND – OR - NOT) ويمكن اعتبار منقي البيانات هو العنصر الالكتروني المناظر للمفتاح الميكانيكي الدوار وكذلك هو دائرة منطقية تختار المعلومات من خطوط المداخل ويكون عدد مداخلها اثنين او اكثر ولها مخرج واحد ومفاتيح تحكم.

### 1 – منتقي البيانات واحد من اثنين 2 TO 1 Multiplexes

الرمزالمنطقي :

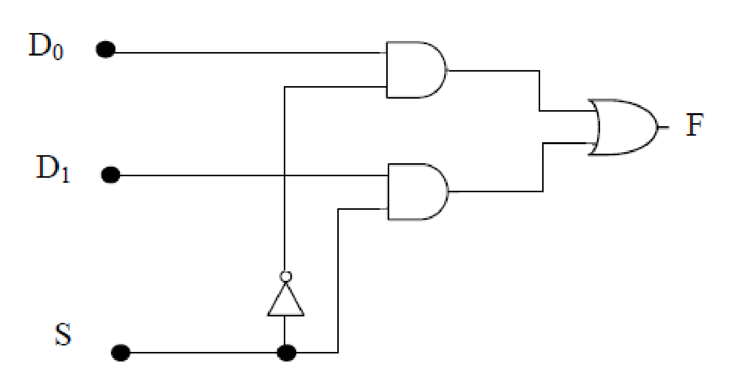





جدول الحقيقة

| S | F |
|---|---|
| 0 | $D_0$ |
| 1 | $D_1$ |

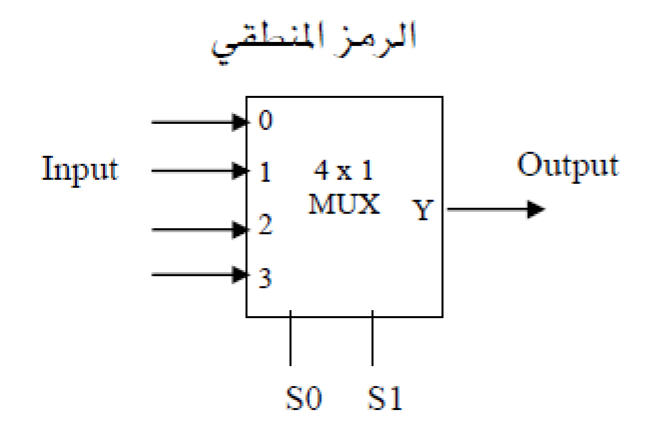
الدائرة المنطقية ل منتقي البيانات واحد من اثنين :



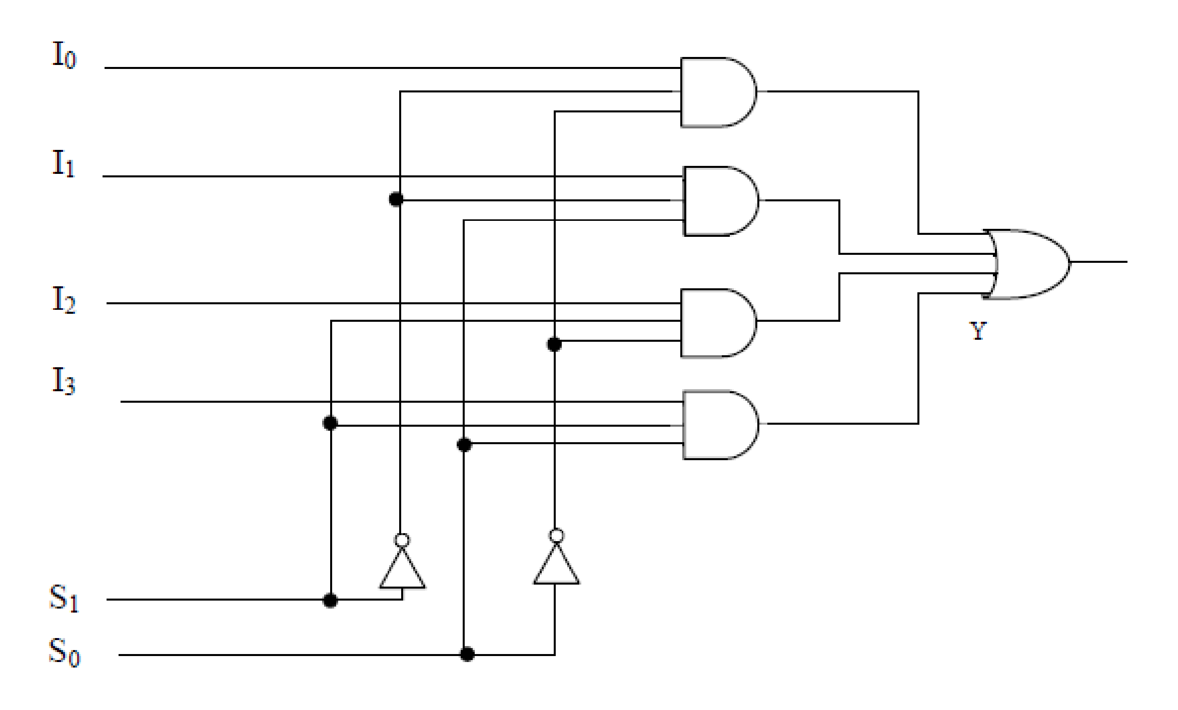

2 - منتقي البيانات واحد من اربعة 4 TO 1 Multiplexes

جدول الحقيقة

| $S_1$ | $S_0$ | Y |
|---|---|---|
| 0 | 0 | $I_0$ |
| 0 | 1 | $I_1$ |
| 1 | 0 | $I_2$ |
| 1 | 1 | $I_3$ |



الدائرة المنطقية المكافئة لها :

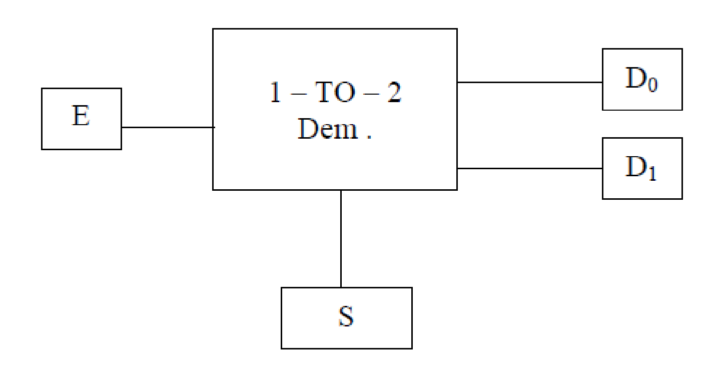



## موزع البيانات Demultipexes

موزع البيانات هو دائرة منطقية لها مدخل واحد يحمل بيانات وعدة مخارج يتم نقل البيانات اليها

1 – موزع البيانات واحد الى اثنين 1 TO Multiplexes 2

الرمز

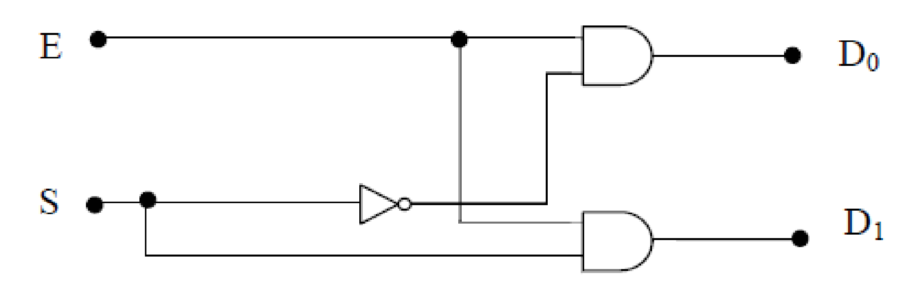


جدول الحقيقة :

| S | $D_0$ | $D_1$ |
|---|---|---|
| 0 | E | 0 |
| 1 | 0 | E |

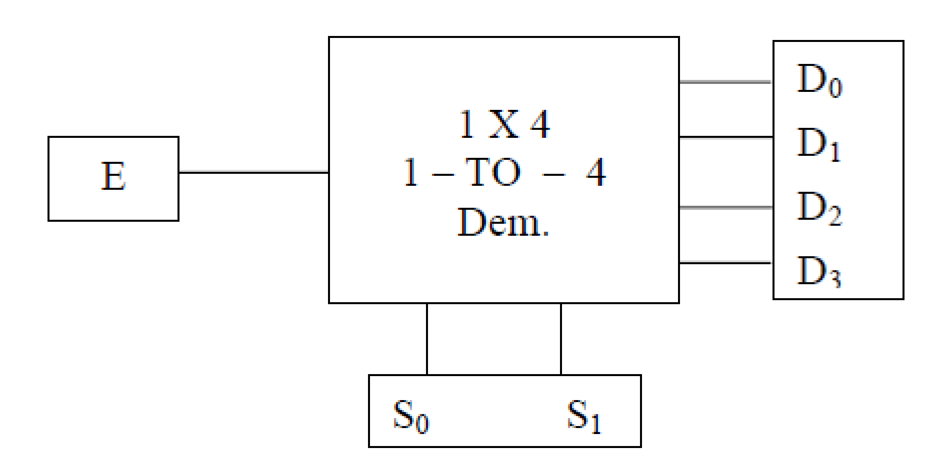
من جدول الحقيقة اعلاه عندما تكون اشارة التحكم S في حالة Logic 0 فان الاشارة تنتقل الى الخرج $D_0$ اما عندما تكون اشارة التحكم في حالة Logic 1 فان الاشارة تنتقل الى الخرج $D_1$ .

الدائرة المنطقية المكافئة لها :




2 - موزع البيانات واحد الى اربعة 1 TO Multiplexes 4

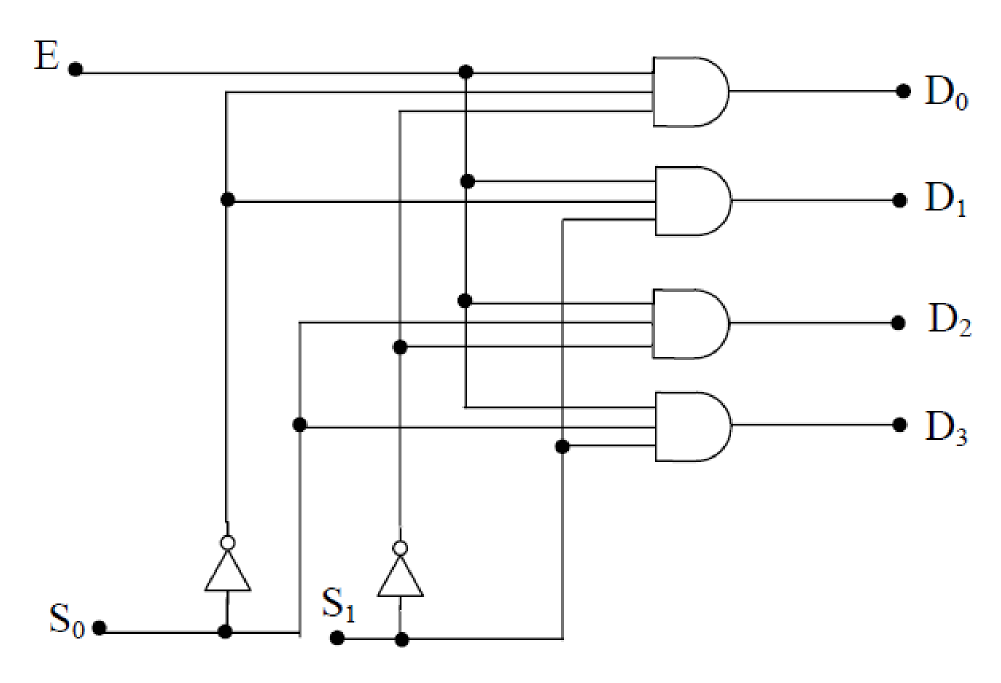
الرمز :



جدول الحقيقة :

| $S_0$ | $S_1$ | $D_0$ | $D_1$ | $D_2$ | $D_3$ |
|---|---|---|---|---|---|
| 0 | 0 | E | 0 | 0 | 0 |
| 0 | 1 | 0 | E | 0 | 0 |
| 1 | 0 | 0 | 0 | E | 0 |
| 1 | 1 | 0 | 0 | 0 | E |

الدائرة المنطقية المكافئة لها :

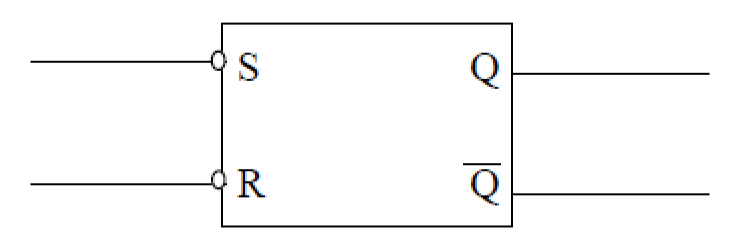


# القلابات Flip-Flops

يتم تصنيف الدوائر المنطقية الى قسمين اساسيين وهم مجموعات البوابات التي صنفناها فيما سبق باسم الدوائر المنطقية التوافقية Combinational Logic Circuits والتي كانت تمثل فيها البوابات المنطقية ركيزة البناء الاساسية . سنقوم الآن بدراسة القسم الثاني وهو الدوائر المنطقية التعاقبية والتي تتميز بخاصية الذاكرة وركيزة البناء الاساسية فيها وهي دائرة القلاب .

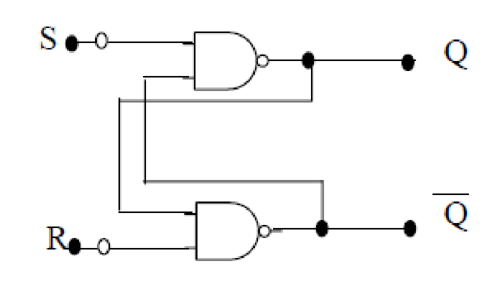
## اولا : القلاب R.S    RSFLIP - FLOP

هو ابسط انواع القلابات يسمى ( القلاب R.S ) ويبين الشكل في الاسفل رمز القلاب R.S ويبين الرمز ان لهذا القلاب مدخلين على اليسار مدخل الوضع في الحالة 1 "S" ومدخل الوضع في الحالة 0 "R" والمدخلان ينشطان بالمستوى المنطقي المنخفض 0 ولذلك تظهر الفقاعتان الصغيرتان على المدخلين S.R وللقلاب مخرجان متتامان , الخرج "Q" يسمى بالخرج الطبيعي وهو المستخدم في العادة والخرج Q هو الخرج المتمم للمخرج الطبيعي .




يمكن بناء قلاب R.S من البوابات المنطقية ويبين الشكل التالي توصيل قلاب R.S مع بوابتين NAND ولاحظ خاصية التغذية المرتدة من مخرج احدى بوابتي NAND الى مدخل البوابة الاخرى ويبين الجدول كيفية تشغيل القلاب :

| وضع التشغيل | المدخلات | | المخرجات | |
|---|---|---|---|---|
| | S | R | Q | $\overline{Q}$ |
| حظر | 0 | 0 | 1 | 1 |
| الوضع في الحالة 1 | 0 | 1 | 1 | 0 |
| الوضع في الحالة 0 | 1 | 0 | 0 | 1 |
| إمساك | 1 | 1 | لا تغير | |



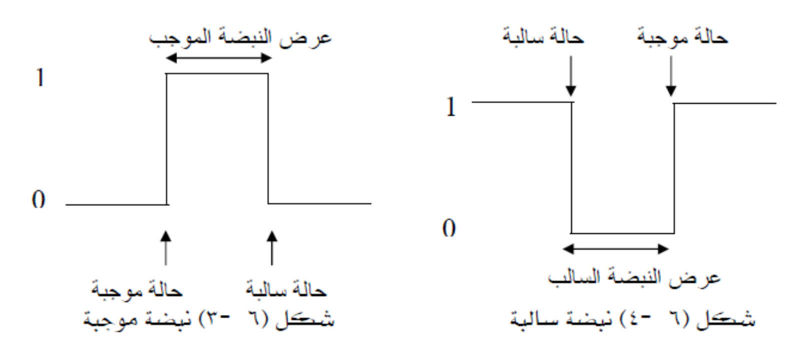

## قدح القلابات FLIP– FLOPS TRIGGERING

في القلابات الغير متزامنة تتغير اشارات الدخل فيها فتؤدي مباشرة الى تغيير حالة الخرج اما القلابات المتزامنة فانها تحتاج الى مدخل قدح (مدخل تزامن Clock) اضافي والذي بدونه لن تعمل هذه القلابات المتزامنة . لذلك يجب عند تشغيل القلابات المتزامنة اعطاء اشارات الدخل اولا ثم اعطاء نبضة قدح (تزامن) على مدخل القدح عند هذه الحالة يتغير الخرج .

انواع نبضات القدح , هناك نوعين من النبضات التي تستخدم لقدح القلابات وهي :

1 – نبضة موجبة : هذه النبضة تكون بدايتها 0 وعند القدح تصعد الى 1 لفترة معينة ثم تعود مرة اخرى من 1 الى 0 انظر الشكل (3-6)

2 – نبضة سالبة : وهذه النبضة تكون بدايتها 1 وعند القدح تهبط الى 0 لفترة معينة ثم تعود مرة اخرى من 0 الى 1 انظر الشكل (4-6)

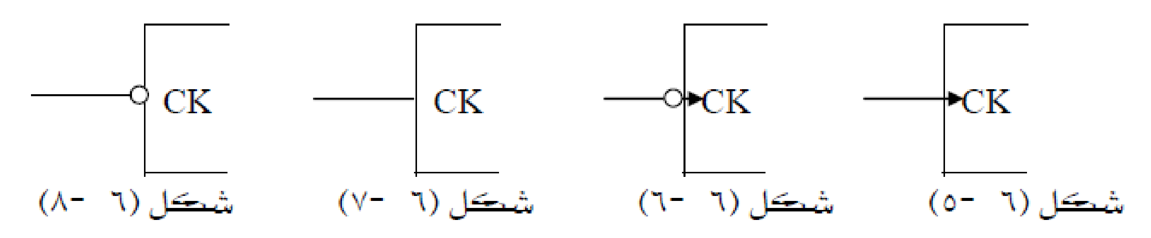



شكل (٦ -٣) نبضة موجبة

شكل (٦ -٤) نبضة سالبة

طرق قدح القلابات المتزامنة , تختلف القلابات في طرق قدحها من قلاب لآخر وهناك اربعة انواع في طرق القدح وهي كالآتي :

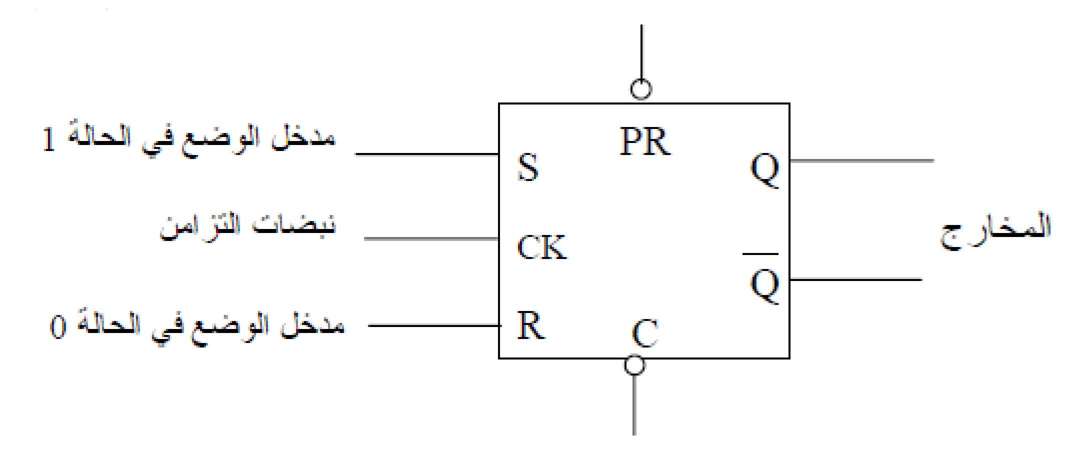
1- نبضة قدح بحافة موجبة , شكل (6-5)
2- نبضة قدح بحافة سالبة , شكل (6-6)
3- نبضة قدح بعرض النبضة الموجبة , شكل (6-7)
4- نبضة قدح بعرض النبضة السالبة , شكل (6-8)




ثانيا : القلاب R.S المتزامن WSYNCHRONOUS R.S FLIP - FLOP

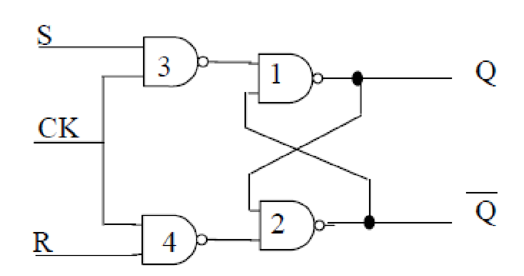
ان قلاب R.S الاساسي عبارة عن شريحة غير متزامنة , فهو لا يعمل وفقا لنبضات تزامن او توقيت بل عندما تنشيط مدخل (مثل المدخل S ) فان الخرج الطبيعي يتم تنشيطه في الحال تماما مثل الدوائر المنطقية التوافقية , يضيف قلاب R.S المتزامن خاصية تزامنية هامة لقلابات R.S ويعمل قلاب R.S وفقا لنبضات تزامن او توقيت.

ويبين الشكل الرمز المنطقي للقلاب :



ويظهرالمدخلان الديناميكيان S (مدخل الوضع في الحالة 1 ) , R (مدخل الوضع في الحالة 0) بالاضافة الى مدخل التزامن الاضافي CK , وكذلك المدخلان الاستاتيكيان PR وهو عند تنشيطه يجعل الخرج الطبيعي HIGH , والمدخل C وهو عند تنشيطه يجعل الخرج الطبيعي LOW ونلاحظ من الرمز المنطقي ان تنشيط المدخلين (PR,C) يتم عن طريق LOW وهما مدخلان غيرمتزامنين لذا عند تنشيط المداخل المتزامنة (R.S) واحد المداخل الغير متزامنة (PR,C) في نفس الوقت فان اولوية التشغيل تكون للمداخل الغير متزامنة لذا يمكن ان نقول ان المداخل الغير متزامنة (الاستاتيكية) اقوى من المداخل المتزامنة (الديناميكية) كما يظهر من المخرجان المعتادان المخرج الطبيعي والمتمم .

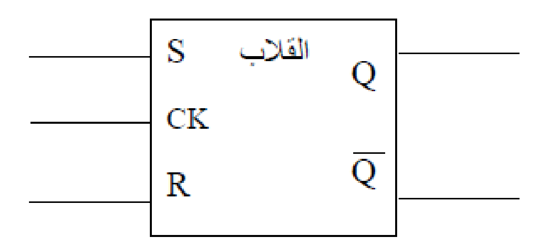
وتوضح الدائرة التالية توصيل قلاب R.S المتزامن باستخدام بوابات NAND بالاضافة الى جدول الحقيقة :




| وضع التشغيل | المدخلات | | | المخرجات | |
|---|---|---|---|---|---|
| | CK | S | R | Q | $\overline{Q}$ |
| الإمساك | ⎍ | 0 | 0 | لا تغيير | |
| وضع الحالة 0 | ⎍ | 0 | 1 | 0 | 1 |
| وضع الحالة 1 | ⎍ | 1 | 0 | 1 | 0 |
| الحظر | ⎍ | 1 | 1 | 1 | 1 |

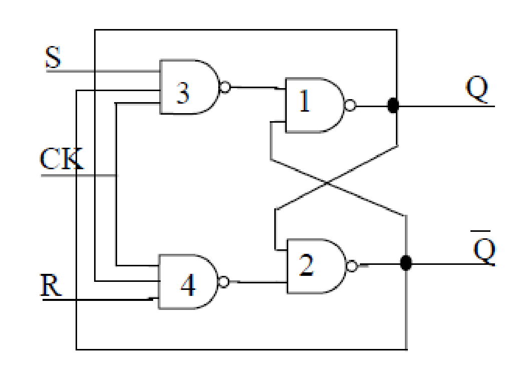
## ثالثا : القلاب J.K    J.K FLIP - FLOP

يبين الشكل التالي الرمز الخاص بقلاب J.K



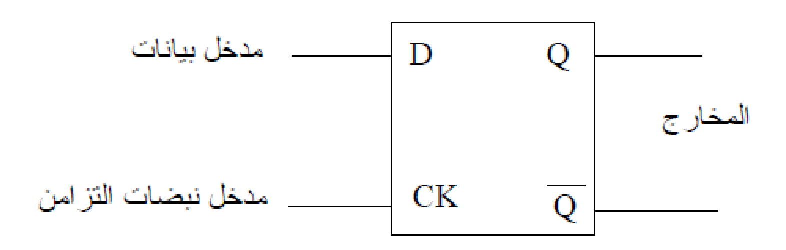

يمكن اعتبار هذا القلاب هو القلاب العام فنلاحظ وجود ثلاثة مداخل متزامنة وخرجين هما الخرج الطبيعي والمتمم , وقد صمم هذا القلاب للتغلب على الوضع المحظور في قلاب R.S المتزامن , ففي حالة تنشيط المدخلين بالوضع المنطقي 1,1 يكون القلاب في وضع تبديل TOGGLE ويوضح الشكل توصيل قلاب J.K باستخدام بوابات NAND وجدول الحقيقة الخاص بتشغيله , كما يمكن اضافة مداخل غير متزامنة لهذا القلاب مثل مدخل PR والذي تنشيطه يجعل الخرج 1 ومدخل CLR عند تنشيطه يجعل الخرج 0 بحيث تعتبر المداخل الغير متزامنة اقوى من المداخل المتزامنة عند التنشيط .

| وضع التشغيل | المدخلات | | | المخرجات | |
|---|---|---|---|---|---|
| | CK | S | R | Q | |
| الإمساك | ⎍ | 0 | 0 | لا تغيير | |
| وضع الحالة 0 | ⎍ | 0 | 1 | 0 | 1 |
| وضع الحالة 1 | ⎍ | 1 | 0 | 1 | 0 |
| الحظر | ⎍ | 1 | 1 | الحالة العكسية | |



رابعا : القلاب D     D FLIP – FLOP

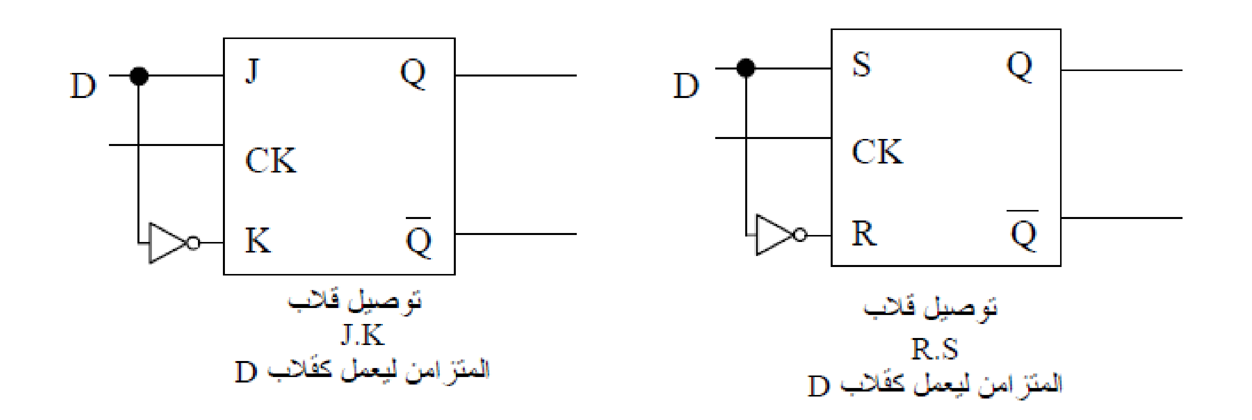
يبين الشكل التالي الرمز المنطقي للقلاب D:




نلاحظ وجود مدخل واحد للبيانات (D) ومدخل التزامن (CK) ويمى كذلك بقلاب التاخير DELAY لان بيانات الدخل تظهر على الخرج بعد نبضة واحدة , ويبين الجدول الحقيقة كيف بعمل القلاب :

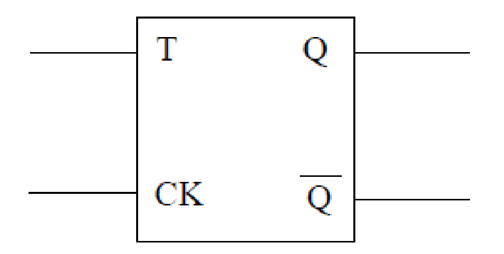
| D.FLIP - FLOP | | | | |
|---|---|---|---|---|
| **CK** | **D** | **Q** | **$\overline{Q}$** | وضع القلاب |
| ⎍ | 0 | 0 | 1 | وضع في الحالة (0) |
| ⎍ | 1 | 1 | 0 | وضع في الحالة (1) |

يمكن بناء القلاب D من القلاب R.S باضافة بوابة NOT على المدخل R كما سنرى في الدائرة بالاسفل وكذلك يمكن بناء القلاب D من القلاب J.K باضافة بوابة NOT على المدخل K , وبذلك يمكن اعتبار القلاب D حالة خاصة من قلابي R.S و J.K المتزامن . تستخدم قلابات D بكثرة في تخرين البيانات ونظرا لهذا الاستخدام فانه يطلق عليه احيانا (قلاب بيانات) كذلك يمكن اضافة مداخل عير متزامنة على القلاب D وهما PR والذي عند تنشيطه يجعل الخرج يساوي 1 و CLR الذي عند نشيطه يجعل الخرج يساوي 0 .

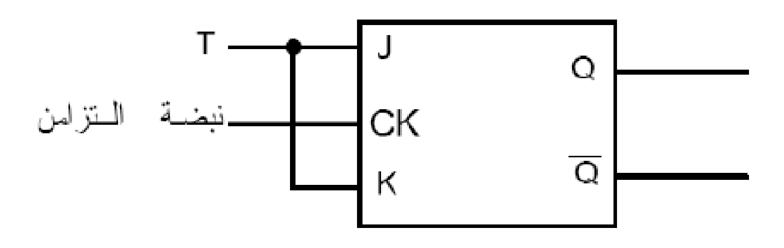



خامسا : القلاب T     T FLIP - FLOP

يبين الشكل التالي الرمز المنطقي للقلاب T :

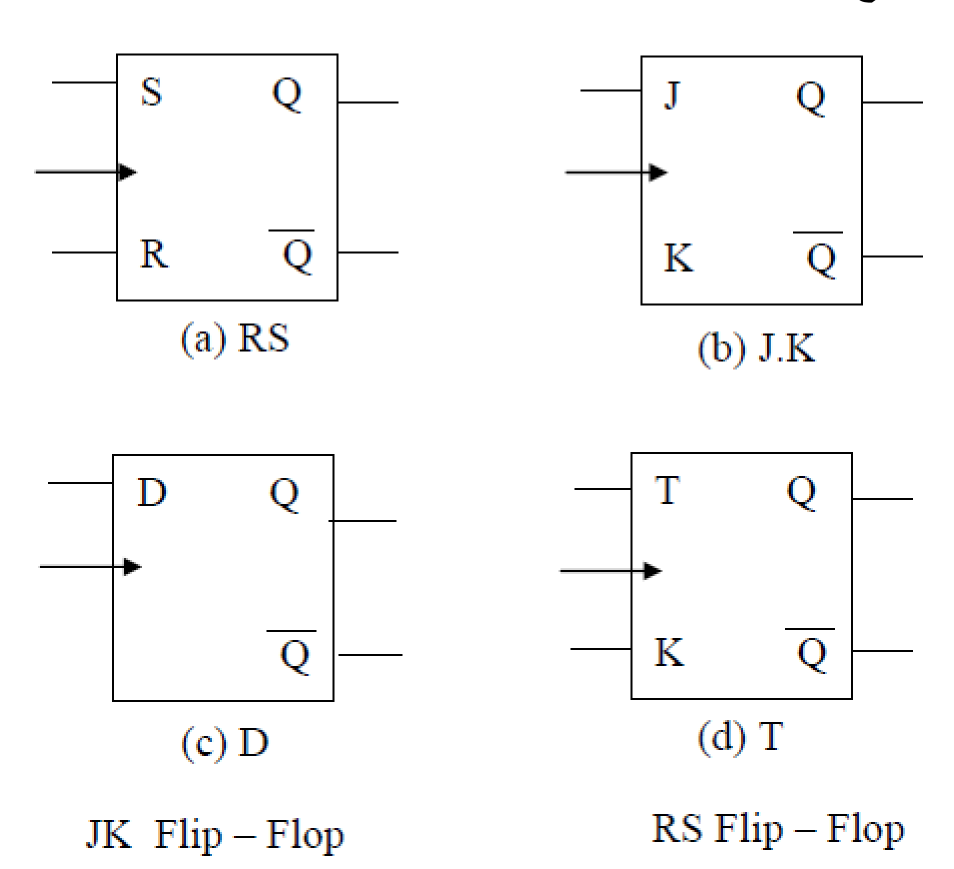



فنلاحظ وجود مدخل واحد T ومدخل التزامن CK ويعتبر القلاب T حالة خاصة من القلاب J.K فعندما يكون الدخل T يساوي 0 منطقيا فان الخرج يكون في حالة امساك اما اذا كان الدخل T يساوي 1 منطقيا فان الخرج يكون في حالة تبديل TOGGLE لذلك يسمى القلاب T بقلاب التبديل وهو يعتبر مقسم للتردد , كذلك يمكن اضافة مداخل غير متزامنة PR,CLR والشكل التالي يوضح كيفية بناء قلاب T من فلاب J.K مرفق معه جدول الحقيقة :



| CK | T | $Q(t+1)$ | وضع التشغيل |
|---|---|---|---|
| ⎍ | 0 | $Q(t)$ | No Change |
| ⎍ | 1 | $\overline{Q}(t)$ | Toggle |

ملخص يوضح الرموز المنطقية للقلابات المتزامنة وجدول الحقيقة الخاص بها :



JK Flip – Flop

| J | K | Q( t - 1) | |
|---|---|---|---|
| 0 | 0 | Q(t) | No change |
| 0 | 1 | 0 | Reset |
| 1 | 0 | 1 | Set |
| 1 | 1 | Q’(t) | Complement |

RS Flip – Flop

| S | R | Q( t + 1) | |
|---|---|---|---|
| 0 | 0 | Q(t) | No change |
| 0 | 1 | 0 | Reset |
| 1 | 0 | 1 | Set |
| 1 | 1 | ? | Unpredictable |

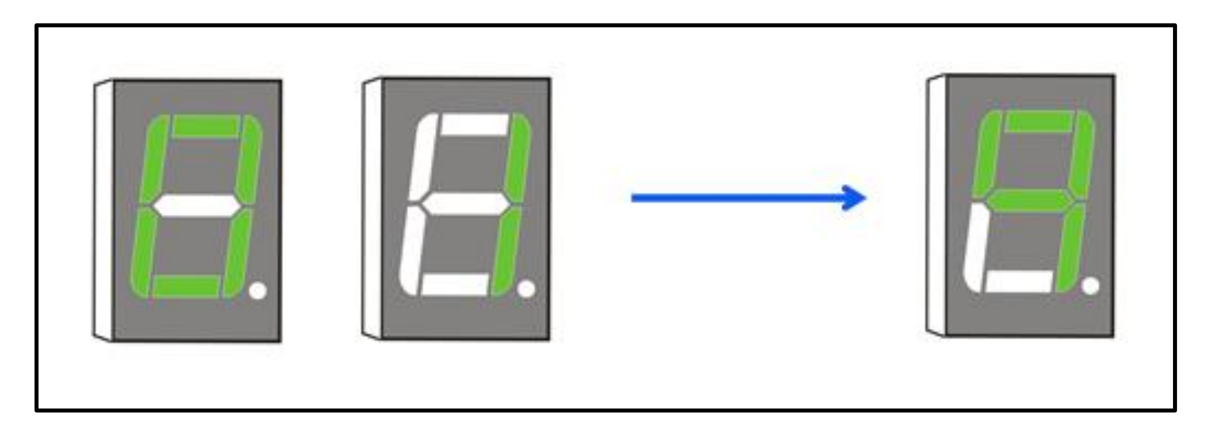
D Flip – Flop

| D | Q(t+1) | |
|---|---|---|
| 0 | 0 | No change |
| 1 | 1 | Set |

T Flip – Flop

| T | Q(t+1) | |
|---|---|---|
| 0 | Q(t) | No change |
| 1 | Q’(t) | Complement |

## العدادات Counters



العداد هو عبارة عن دائرة منطقية تعاقبية تعطي خرجا له تسلسل منطقي معين , وتعتبر العدادات من اعظم الدوائر المنطقية واكثرها استخداما فالتوقيت امر بديهي وذات اهمية في الدوائر الرقمية , وعادة يتم بناء العداد اساسيات من القلاب J.K او قلاب T .
وللعدادات الرقمية الخصائص التالية :

- اقصى عدد يستطيع العداد احصاؤه .
- طريقة العد تصاعديا (UP) او تنازليا (DOWN) .

تستطيع بعض العدادات العد بالاتجاهين حسب ما تقرره اشارة التحكم في الداخل , ويتم ذلك اما بتطبيق اشارة على المدخل U/D لتقرر اتجاه العد , او باستخدام مدخلي ساعة احدهما للعد التصاعدي والآخر للعد التنازلي .

- التشغيل المتزامن (synchronous) او غير الغير متزامن (Asynchronous)
- الحجم SIZE

تاتي العدادات 4Bit بصيغة BCD ( تقسيم على 10 ) او بالصيغة الثنائية وهناك عدادات اكبر حجما تصل الى 24 Bit كما ان هناك عدادات قسمت بالنسبة لعدد صحيح n ويمكننا دائما توصيل العدادات على التعاقب بما في ذلك الانواع المتزامنة للحصول على مراحل اطور .

- الساعة Clocking

من الفروق الهامة بين العدادات نذكر كونها عدادات متموجة ripple ام متزامنة synchronous , في العدادات المتزامنة تقوم الساعة بقدح جميع القلابات سوية بينما

في العدادات المتموجة يتم قدح كل مرحلة بخرج المرحلة السابقة , والعدادات المتموجة ابطأ من العدادات المتزامنة بسبب تراكم ازمان تاخير الانتشار . ويتم قدح العدادات المتموجة بالحافة الهابطة لتهسيل عملية توسيعها , اما العدادات المتزامنة فتقدح بالحافة الصاعدة .

- التحميل والتصفير Load and clear

ان معظم العدادات تحتوي على مدخل معطيات بحيث يمكن تحميلها بقيمة مسبقة قبل بدء العد , وهذ مفيد اذا اردنا تصميم عداد يقسم على قيمة معينة n .

وعملية التصفير clear او (RESET) هي شكل من اشكال التحميل , والغالبية من العدادات تحتوي تابع تصفيري قصري .

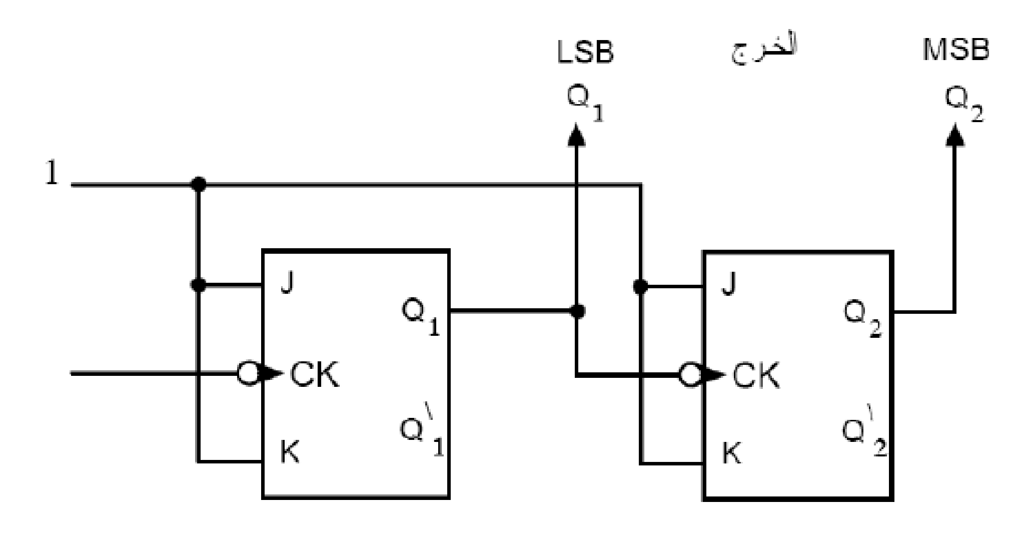
اولا : العدادات الغير متزامنة ( المتموجة ) ASYNCHRONOUS

هي عدادات يتم فيها توصيل نبضة التزامن CK للقلاب الاول ويقدح القلاب الثاني من خرج القلاب الاول وهكذا .

وتقسم العدادات الغير متزامنة الى :

1 – العدادات التصاعدية UP-Counters

أ – عداد تصاعدي ذو معامل (4) باستخدام قلابات J.K



يبين الشكل في الاعلى دائرة عداد تصاعدي غير متزامن ذو معامل 4 اي له اربعة حالات عد (يبدء من 0 وينتهي 3 عشري ) ويتكون هذا العداد من قلابي J.K ومدخلي k و J لكل قلاب موصلة بالمستوى المنطقي 1 .

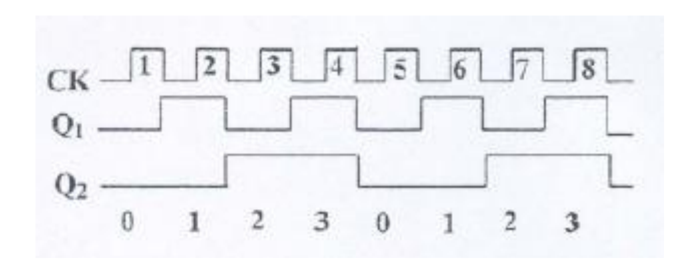
ونلاحظ ان كلا القلابين يعملان عند الحافة السالبة لنبضة التزامن ودخل التزامن للقلاب الثاني موصل بالخرج المادي Q للقلاب الاول , ومخارج العداد في النهاية هما الخرج العادي كما هو موضح Q1 و 2Q .

والجدول التالي يوضح جدول الحقيقة لتشغيل هذا العداد فالقلاب الاول يكون في حالة تبدل مستمرة عند الحافة السالبة لنبضات التزامن والقلاب الثاني يكون في حالة تبدل عند الحافة السالبة للنبضة الثانية لنبضات التزامن , وسوف يعد العداد من 0 الى 3 وعد الاستمرار في نبضات التزامن فان العداد يعيد العد مرة اخرى من 0 الى 3 وهكذا.

جدول الحقيقة لعداد تصاعدي ذو معامل (4) :

| CLK NO. | O/P | | المكافيء العشري |
|---|---|---|---|
| | $Q_2$ | $Q_1$ | |
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 2 | 1 | 0 | 2 |
| 3 | 1 | 1 | 3 |
| 4 | 0 | 0 | 0 |
| 5 | 0 | 1 | 1 |
| 6 | 1 | 0 | 2 |
| 7 | 1 | 1 | 3 |

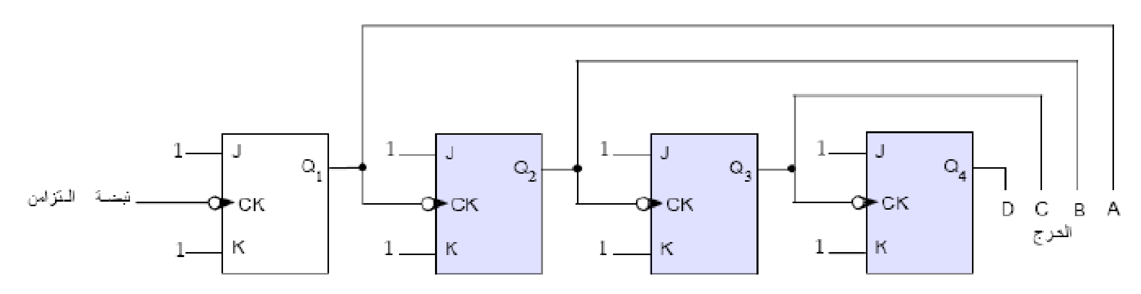
ويبين الشكل التالي الخرج الموجي لهذا العداد لثمان نبضات تزامن حيث نرى من هذا الخرج ان العداد يعتبر مجزئ او مقسم للتردد بحيث ان عدد نبضات الخرج للقلاب الاول Q1 يساوي اربعة نبضات وعدد نبضات الخرج للقلاب الثاني Q2 يساوي نبضتان اي ان القلاب الاول يقسم على 2 والقلاب الثاني يقسم على 4 .




حسنا سنقوم الآن في تصميم عداد تصاعدي ذو معامل (16) وذلك باستخدام قلابات J.K مع توضيح حالات العد باستخدام جدول الحقيقة ونرسم الشكل الموجي للخرج.

كيف نقوم بالتصميم ؟

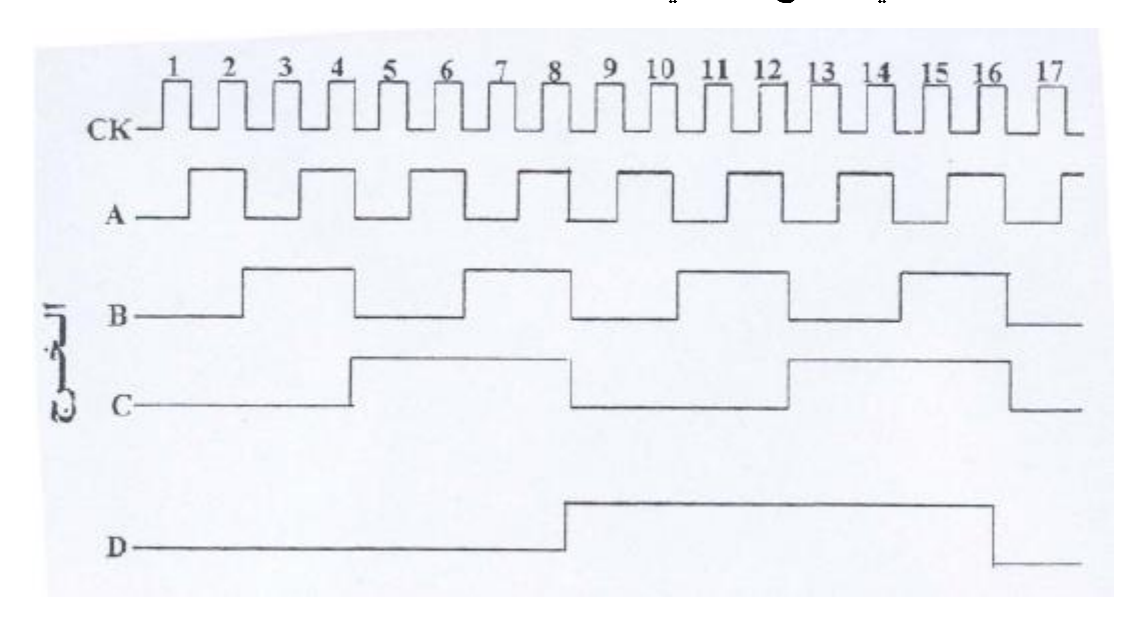
ان عداد ذو معامل 16 هو عداد يعد من 0 الى 15 ويمكن استنتاج عدد القلابات المستخدمة عن طريق العلاقة $2^m=16$ حيث m عدد القلابات , وبالتالي عندما تكون m=4 فهذا يعني $2^4=16$ اي ان عدد القلابات هو اربع قلابات J.K وتكون الدائرة المنطقية كالتالي :




وجدول الحقيقة لعداد تصاعدي ذو معامل (16) :

| العدد العشري | العدد الثنائي | | | | العدد العشري | العدد الثنائي | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 8 | 4 | 2 | 1 | | 8 | 4 | 2 | 1 |
| | D | C | B | A | | D | C | B | A |
| 0 | 0 | 0 | 0 | 0 | 8 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 9 | 1 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 | 10 | 1 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 | 11 | 1 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 | 12 | 1 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 13 | 1 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 0 | 14 | 1 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 | 15 | 1 | 1 | 1 | 1 |

ويبين الشكل التالي الخرج الموجي لهذا العداد :



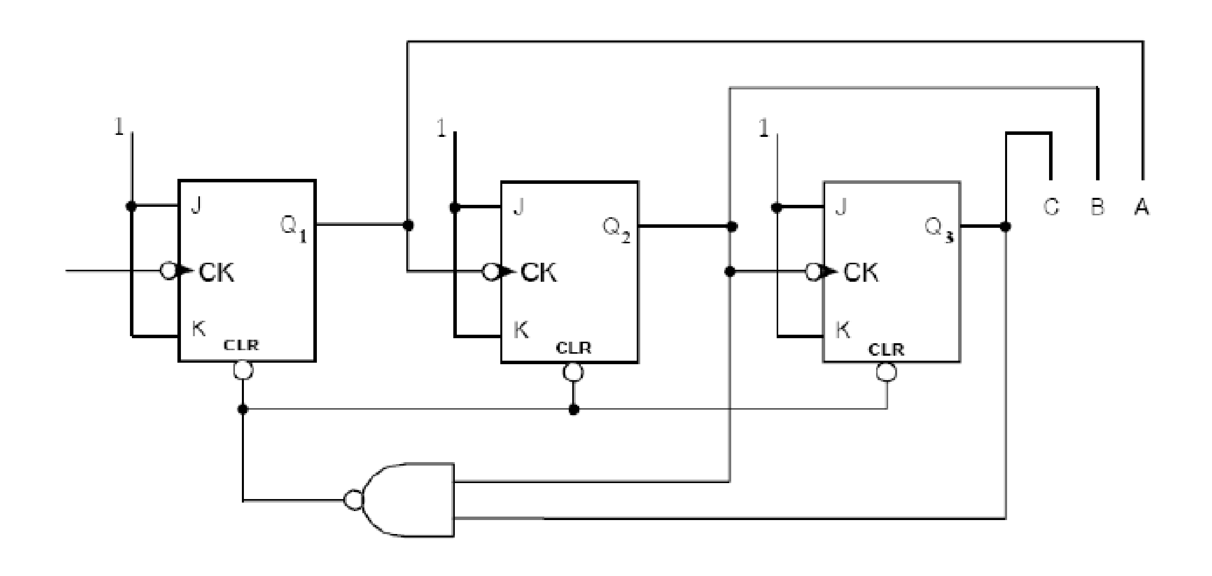

ب – عداد تصاعدي ذو معامل (n) :

عندما نريد تصميم عداد ذو معامل n فاننا نطبق القاعدة ( $2^m \geq n$ ) حيث ان

m : عدد القلابات , n : معامل العداد

فمثلا عندما نريد تصميم عداد ذو معامل 6 اي له ست حالات عد من 0 الى 5 نطبق القاعدة ( $2^3 \geq 6$ ) ولانه لا يوجد عدد n يعطينا $2^m$ تساوي 6 ناخذ الاكبر 8 ولكن هذا 8 تعني 8 حالات اي من 0 الى 7 لذلك فاننا نحتاج الى ثلاثة قلابات J.K

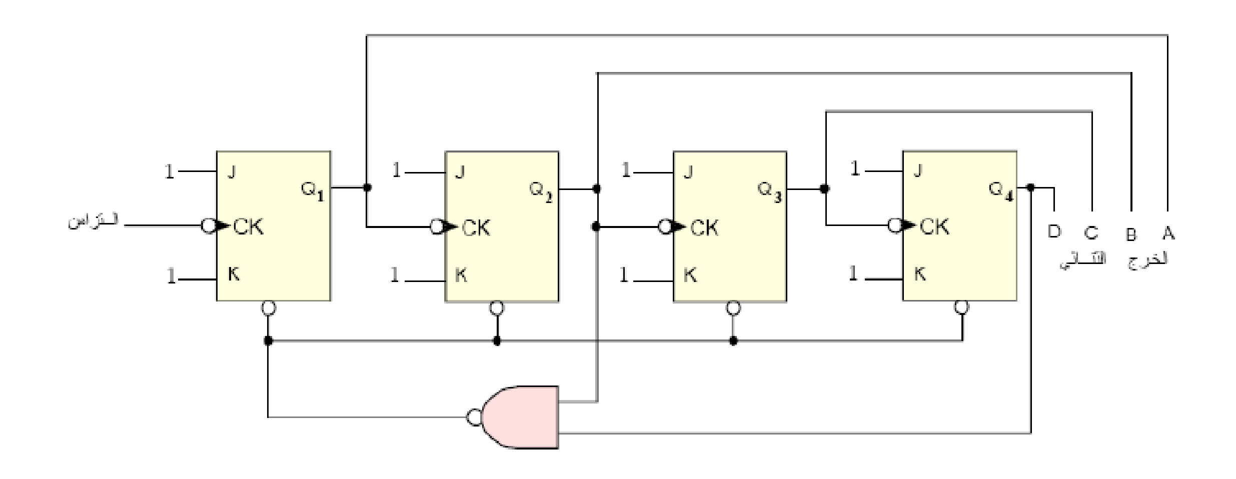
ولتكن A,B,C وكذلك نحتاج الى بوابة NAND تكون مداخلها من المكافئ الثنائي للرقم للرقم العشري (6) اي بوابة NAND دخلها من خرج القلابان C,B وخرج البوابة يكون دخل لمدخل المسح CLR للقلابات , لذا فانه عندما يعد العداد خمسة والذي يكافؤه ثنائيا سوف ينتقل العداد لعد العدد 6 الذي يكافؤه ثنائيا وهذا ينشط بوابة NAND لذا فان خرجها سيكون صفر , وهذا بدوره ينشط مدخل المسح مما يؤدي الى تصفير جميع مخارج القلابات وتبدأ العد من جديد (000) ولا تعد (110) .ويوضح الشكل التالي بناء هذا العداد :




ملاحظة : اذا كانت مداخل المسح للقلابات تنشط بالصفر نستخدم بوابة NAND اما اذا كانت تنشط بالواحد نستخدم بوابة AND .

- العداد العشري Decimal counter

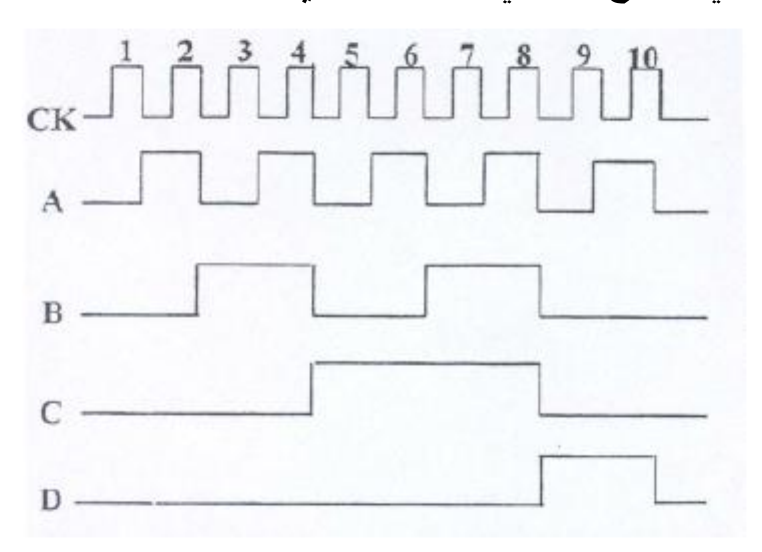
يعبر العداد العشري اكثر انواع العدادات انتشارا بفضل تطبيقاته واستخداماته الكثيرة وهو عداد ذو معامل عشرة اي ان العداد يعد من 0 الى 9 عشري , اي من 0000 الى 1001 ثنائي , ويتكون العداد من اربعة قلابات J.K وبوابة NAND , ويتلخص عمل هذا العداد انه عندما ينتهي العداد من عد العدد تسعة ويبدأ في العدد العشرة والذي يكافئ ثنائيا وهذا يعني لان الخرجين (B=1 , D=1) هما دخلين لبوابة NAND وخرج البوابة ينشط مدخل المسح CLR للقلابات الاربعة , وهذا يجعل جميع القلابات تقوم بعملية المسح لمخارجها لتساوي صفرا وليبدأ العداد من جديد , ويوضح الشكل التالي بناء هذا العداد :



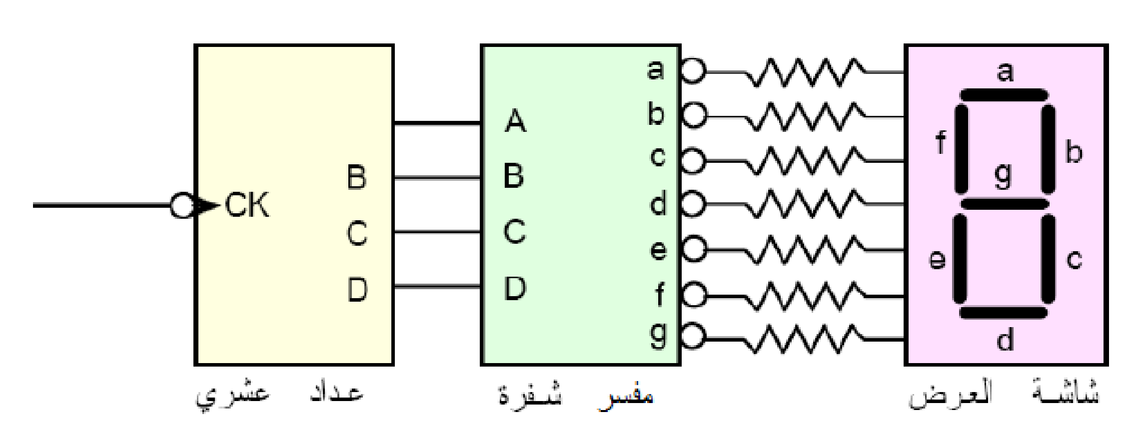

جدول الحقيقة للعداد العشري :

| CLK NO. | O/P المكافئ الثنائي | | | | المكافئ العشري |
|---|---|---|---|---|---|
| | D | C | B | A | |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 2 |
| 3 | 0 | 0 | 1 | 1 | 3 |
| 4 | 0 | 1 | 0 | 0 | 4 |
| 5 | 0 | 1 | 0 | 1 | 5 |
| 6 | 0 | 1 | 1 | 0 | 6 |
| 7 | 0 | 1 | 1 | 1 | 7 |
| 8 | 1 | 0 | 0 | 0 | 8 |
| 9 | 1 | 0 | 0 | 1 | 9 |
| 10 | 0 | 0 | 0 | 0 | 0 |

ويبين الشكل التالي الخرج الموجي للعداد العشري :




الشكل التالي يوضح توصل العداد العشري مع مفسر الشفرة (Decoder) وشاشة عرض الاجزاء السبعة (Seven Segments) :



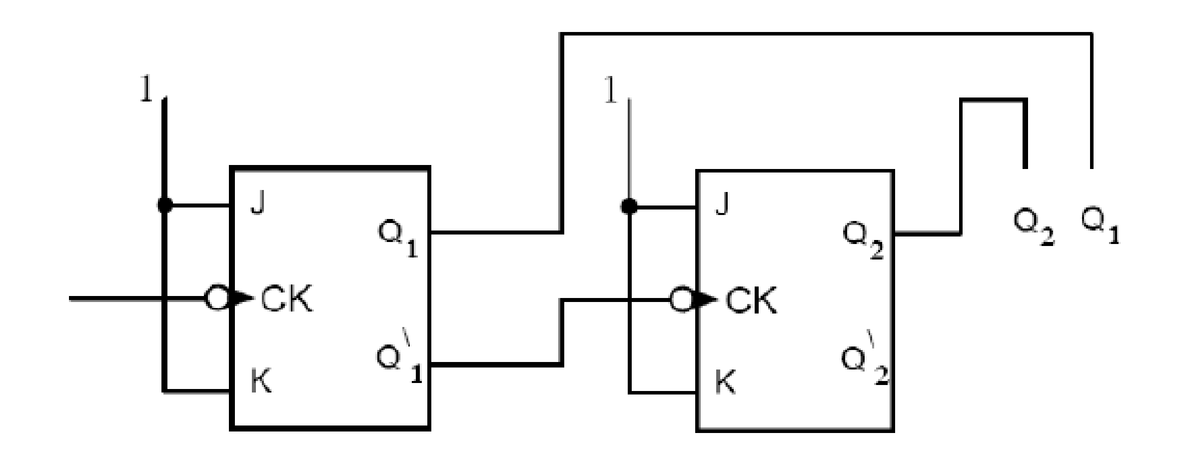

والجدول التالي يبين عمل الدائرة السابقة , مع العلم ان شاشة العرض ذات سبع قطع هي من نوع مشترك الآنود يكون تنشيط الشاشة عند المستوى المنطقي 0 ولهذا تظهر الفقاعات في مخارج مفسر الشفرة :

| CLK NO. | خرج العداد | | | | خرج مفسر الشفرة | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | D | C | B | A | a | b | c | d | e | f | g |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 4 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 6 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

## 2 – العدادات التنازلية Down Counters

أ – عداد تنازلي ذو معامل (4) باستخدام قلابات J.K :

يختلف العداد التنازلي عن العداد التصاعدي في تسلسل العد حيث يبدأ العد التنازلي في العدد من اقصى قيمة ويبدأ في التنازل , ويبين الشكل التالي عداد تنازلي ذو اربع حالات عد بحيث يعد من 3 الى 0 عشري :

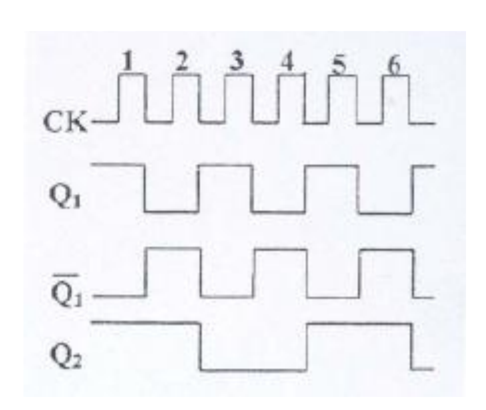


ونلاحظ من المخطط ان هذا العداد يتكون من قلابي J.K ومدخلي J,K لكل قلاب موصلة بالواحد المنطقي ونلاحظ ان مدخل التزامن CK لكلا القلابين يعملان عند الحافة السالبة لنبضة التزامن, ومدخل التزامن للقلاب الثاني موصول بالخرج المتمم $Q1^-$ للقلاب الاول , ومخارج العداد تكون من الخرج العادي للقلابين Q1,Q2 .

والشكل التالي يوضح جدول الحقيقة لتشغيل هذا العداد , فحالة البداية للعداد التنازلي تكون جميع المخارج للعداد في المستوى العالي ( اي اقصى قيم للعدد ) ثم يبدأ العداد في التنازل فالعداد التنازلي ذو معامل 4 سوف يعد من ثلاثة الى صفر وعند الاستمرار في نبضات التزامن فان العداد سوف يعيد العد مرة اخرى من ثلاثة الى صفر وهكذا .

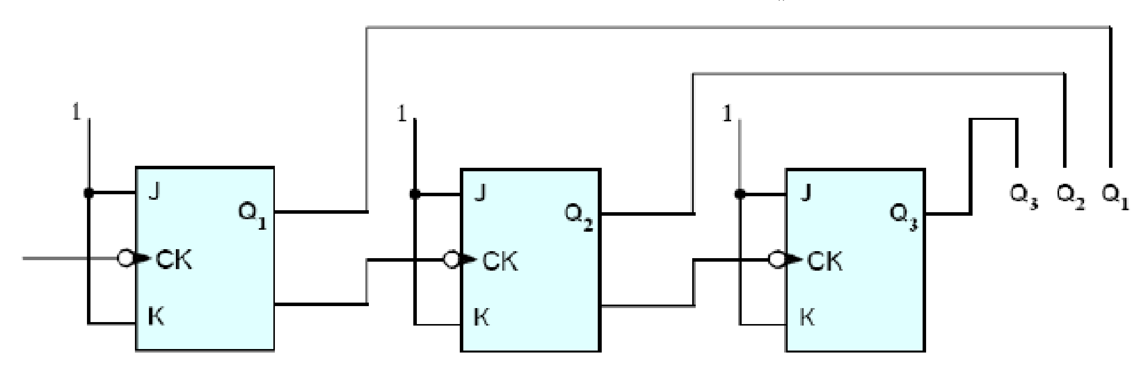
| CLK NO. | O/P | | |
|---|---|---|---|
| | المكافيء الثنائي | | المكافيء العشري |
| | $Q_2$ | $Q_1$ | |
| 0 | 1 | 1 | 3 |
| 1 | 1 | 0 | 2 |
| 2 | 0 | 1 | 1 |
| 3 | 0 | 0 | 0 |
| 4 | 1 | 1 | 3 |

ويبين الشكل التالي سلوك هذا العداد , فالقلاب الاول يكون في حالة تبديل مستمرة عند كل حافة سالبة لنبضات التزامن , والقلاب الثاني يكون مدخل التزامن له هو الخرج المتمم للقلاب الاول $Q1^-$ وبالتالي فان القلاب الثاني سوف يكون في حالة تبديل مستمرة عند كل حافة سالبة ل $Q1^-$




ب – عداد تنازلي متموج (غير مستقر ) ذو معامل (8) باستخدام قلابات J.K

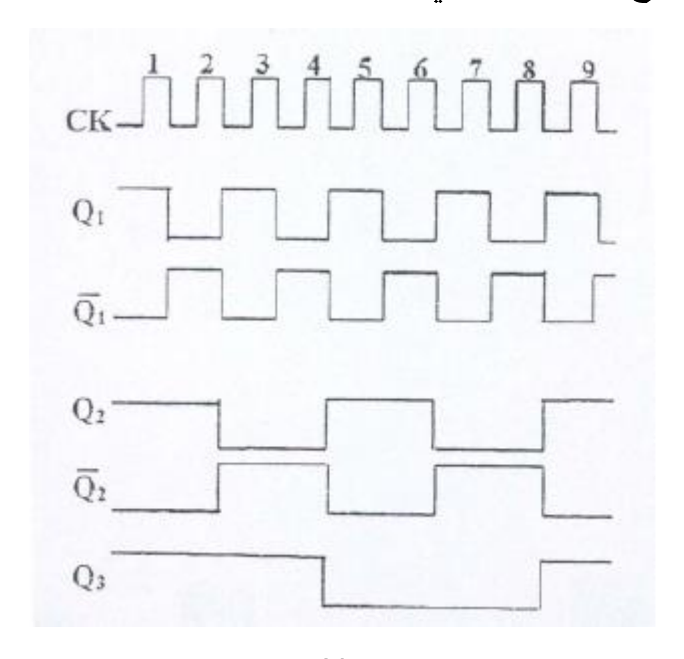
عداد متموج تنازلي ذو معامل (8) اي انه يعد من 7 الى 0 ولاستنتاج عدد القلابات المستخدمة عن طريق العلاقة ($2^3=8$) وبالتالي فاننا نحتاج الى ثلاثة قلابات J.K وسيكون شكل الدائرة كالتالي :



والشكل التالي يوضح جدول الحقيقة للعداد :

| CLK NO. | O/P | | | |
|---|---|---|---|---|
| | المكافيء الثنائي | | | المكافيء العشري |
| | $Q_1$ | $Q_2$ | $Q_3$ | |
| 0 | 1 | 1 | 1 | 7 |
| 1 | 1 | 1 | 0 | 6 |
| 2 | 1 | 0 | 1 | 5 |
| 3 | 1 | 0 | 0 | 4 |
| 4 | 0 | 1 | 1 | 3 |
| 5 | 0 | 1 | 0 | 2 |
| 6 | 0 | 0 | 1 | 1 |
| 7 | 0 | 0 | 0 | 0 |
| 8 | 1 | 1 | 1 | 7 |
| 9 | 1 | 1 | 0 | 6 |

والشكل التالي يوضح الشكل الموجي لهذا العداد :

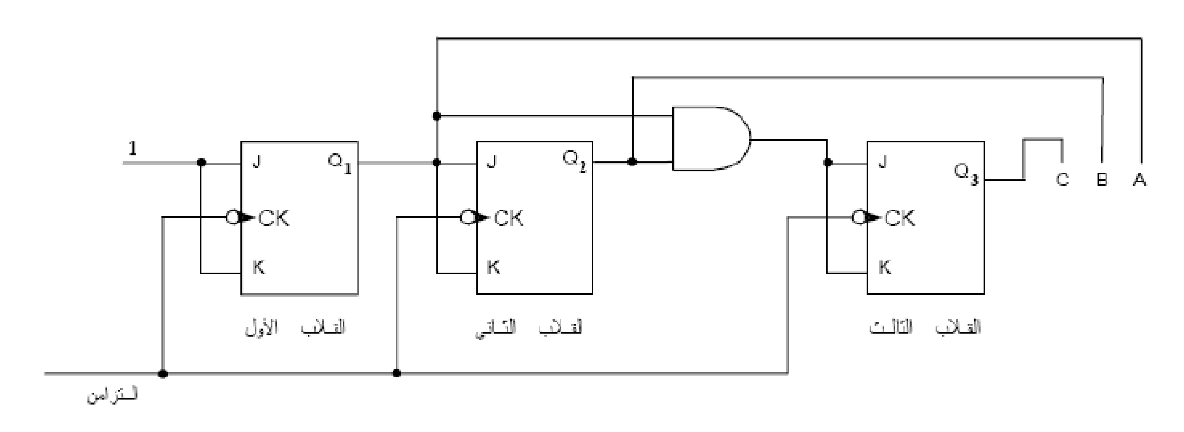


ثانيا : العدادات المتزامنة Synchronous Counters

هي عدادات يتم توصيل مدخل التزامن CK لجميع القلابات في نفس الوقت ( توصيل توازي ) لحل مشكلة التأخير الزمني الناتج في العدادات غير المتزامنة ونسمي هذه النوعية من العدادات بعدادات التوازي Parallel Counters .

1 – عداد تصاعدي متزامن ذو معامل (8) :

يقوم هذا العداد بعد الارقام من 0 الى 7 وبالتالي سوف نحتاج الى ثلاثة قلابات J.K لتصميم هذا العداد , والشكل التالي يوضح مخطط الدائرة :

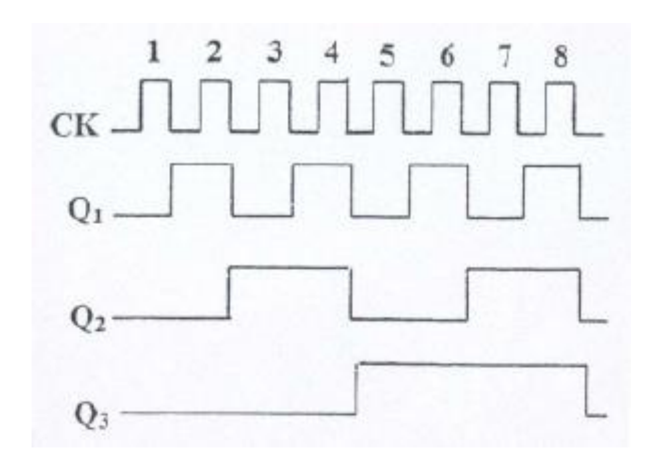



ويلاحظ من الشكل السابق ما يلي :

أ – القلاب الاول يكون دائما في وضع التبديل Toggle او حالة تخرين No Change تبعا لخرج القلاب الاول .

ب – يستخدم خرج القلاب الاول والقلاب الثاني كدخل لبوابة AND وهي تتحكم في تشغيل القلاب الثالث , فعندما يتم تنشيط هذه البوابة عن طريق المستوى المنطقي 1 عند كل من A,B فان القلاب الثالث يصبح في وضع تبديل , وعندما يمنع تنشيط البوابة فان القلاب الثالث يصبح في وضع تخرين .

ويوضح الشكل التالي الشكل الموجي للخرج :

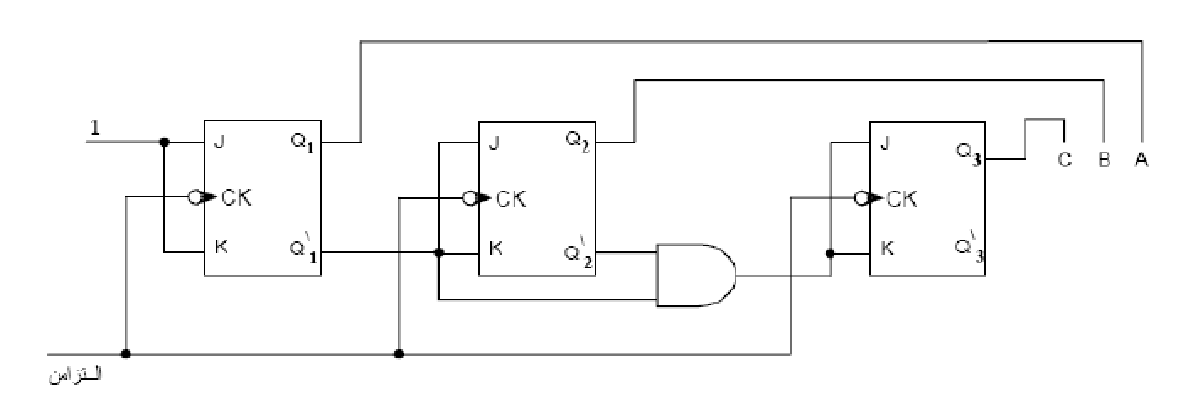



والشكل التالي يوضح جدول الحقيقة للعداد :

| CLK NO. | O/P | | | المكافىء العشري |
|---|---|---|---|---|
| | C | B | A | |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 2 | 0 | 1 | 0 | 2 |
| 3 | 0 | 1 | 1 | 3 |
| 4 | 1 | 0 | 0 | 4 |
| 5 | 1 | 0 | 1 | 5 |
| 6 | 1 | 1 | 0 | 6 |
| 7 | 1 | 1 | 1 | 7 |
| 8 | 0 | 0 | 0 | 0 |

## 2 - عداد تنازلي متزامن ذو معامل (8) :

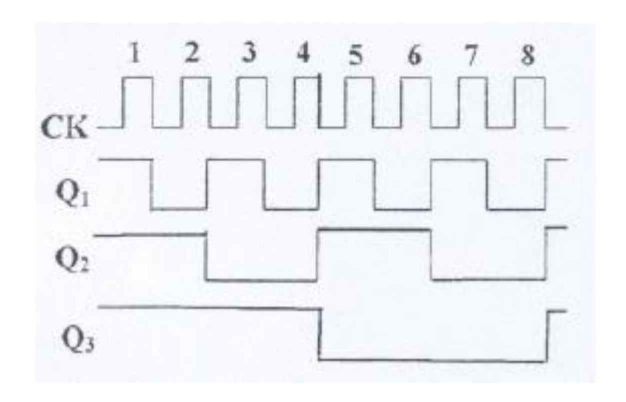
يوضح الشكل التالي رسم الدائرة المنطقية لعداد توازي تنازلي ذو ثلاثة ارقام ثنائية اي معامل 8 ونلاحظ انه قد وصلت مداخل التزامن CK في نفس الوقت لجميع القلابات , ولكن الفرق الوحيد هو ان تشغيل العداد التنازلي نستخدم فيه الخرج المتمم للقلابات في عملية التشغيل .




وجدول الحقيقة :

| CLK NO. | O/P | | | المكافىء العشري |
|---|---|---|---|---|
| | C | B | A | |
| 0 | 1 | 1 | 1 | 7 |
| 1 | 1 | 1 | 0 | 6 |
| 2 | 1 | 0 | 1 | 5 |
| 3 | 1 | 0 | 0 | 4 |
| 4 | 0 | 1 | 1 | 3 |
| 5 | 0 | 1 | 0 | 2 |
| 6 | 0 | 0 | 1 | 1 |
| 7 | 0 | 0 | 0 | 0 |
| 8 | 1 | 1 | 1 | 7 |

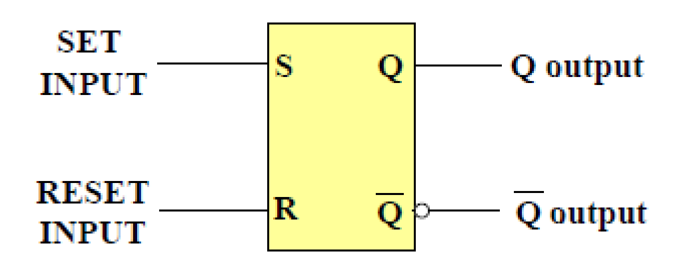
والشكل الموجي للخرج :




## المساكات او المنزلقات Latches

دائرة المساك هي نوع من عناصر التخزين ثنائية الاستقرار والتي عادة ما توضع في تصنيف منفصل عن دوائر القلابات . والمساكات من حيث طبيعة العمل تشبه دوائر القلابات لانها عنصر ثنائي الاستقرار يمكن وضعه في إحدى حالتي الاستقرار بواسطة نظام التغذية الخلفية والذي فيه يوصل الخرج خلفيا الى الدخل المعاكس . والفرق الرئيسي بين المساكات والقلابات هو في الطريقة المستخدمة لتغيير حالتي الاستقرار فقط .

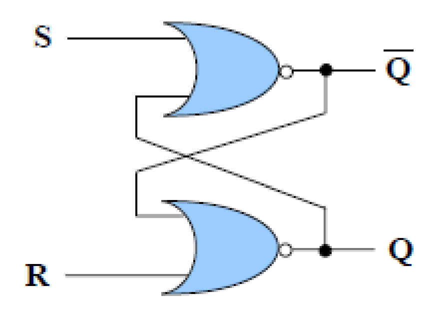
والمساك (Latch) هو نوع من المهتز متعدد التوافقيات ثنائي الاستقرار ( Bistable Multivibrator) . يوضح الشكل التالي الرمز المنطقي لدائرة المساك من النوع S.R ومنه يتضح وجود مدخلين يرمز لاحدهما بالرمز S وهو يسمى بالمدخل الفعال او مدخل الوضع في الحالة 1 (Set Input) ويرمز للآخر بالرمز R ويعرف بالمدخل الغير فعال او مدخل الوضع في الحالة 0 (Reset Input) كما يوجد لها مخرجان يرمز لاحدهما Q ويعرف بالمخرج الطبيعي كما درسنا سابقا والرمز الاخر $Q^-$ ويعرف بالمخرج المتمم .




ويقال ان دائرة المساك في حالة فعالة او نشطة (Set Condition) عندما يكون المخرج المتمم يساوي 0 والمخرج الطبيعي يساوي 1 ويقال عنه في حالة غير فعالة او خاملة (Reset Condition) عندما يكون المخرج المتمم يساوي 1 والمخرج الطبيعي يساوي 0 ومن التعريف الاساسي للمساك نجد انه عنما نؤثر على المدخل S بالمستوى المنطقي 1 يكون المستوى المنطقي للخرج Q=1 . وفي نفس الوقت يكون المتسوى المنطقي للخرج المتمم 0 .

واذا اثرنا على الدخل R بالمستوى المنطقي 0 اي الحالة الغير فعالة بينما يكون مخرج المتمم يساوي 1 , اما اذا اثرنا على S,R بالمستوى المنطقي 1 في نفس الوقت فان مستوى الخرج المنطقي يصبح غير محدد وغير معروف (unpredictable) ويجب تفادي ذلك حتى نتنجب الاخلال بدائرة المساك .

ويمكن بناء دائرة المساك S.R من بوابتي NOR باستخدام خاصية التغذية الخلفية المرتدة من مخرج احدى البوابتين الى مدخل البوابة الاخرى كما هو موضح في الشكل:




دائرة المساك S-R ذو المدخلات الفعالة العالية

ونظرا لان المستوى المنطقي الفعال لبوابة NOR هو 1 ( اي مستوى الدخل الذي يحدث عنده تغير في حالة الخرج ) لذا فان جدول الحقيقة لدائرة المساك في هذه الحالة

يأخذ الصورة الموضحة في الجدول التالي وتسمى الدائرة في هذه الحالة بدائرة المساك ذات المدخلات الفعالة العالية (Active High Inputs) :

| المدخلات | | الخرج | وضع التشغيل |
|---|---|---|---|
| S | R | Q | (Mode of Operation) |
| ٠ | ٠ | Q. | وضع الإمساك (عدم التغير) No Change |
| ٠ | ١ | ٠ | الوضع الغير فعال Latch RESETS |
| ١ | ٠ | ١ | الوضع الفعال Latch SETS |
| ١ | ١ | ? | وضع الخطر أو وضع غير مسموح به Invalid condition |

جدول الحقيقة لدائرة المساك S-R ذات المدخلات العالية

وبالنظرالى جدول الحقيقة الموضح يمكننا ملاحظة الآتي :

1 – عند وجود المستوى المنطقي 0 على المدخلين S,R في نفس الوقت لا تتغير حالة المساك اي تظل قيمة Q (السطر الاول في جدول الحقيقة) ويعرف هذا الوضع بوضع الامساك او عدم التغيير .

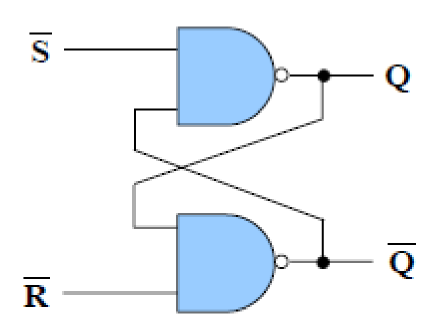
2 – عندما يتغير المستوى المنطقي على الدخل R من 0 الى 1 يتغير المستوى المنطقي للخرج Q الى 0 اي ان Q=0 (الحالة الغير فعالة) كما في السطر الثاني من الجدول اما اذا كان الخرج Q=0 اصلا فيظل كما هو دون تغيير .

3 – عندما يتغير المستوى المنطقي على الدخل S من 0 الى 1 تتغير قيمة المستوى المنطقي على الخرج Q من 0 الى 1 اي ان الخرج الفعال يساوي1 كما في السطر الثالث من الجدول اما اذا الخرج Q اصلا يساوي واحد فيظل كما هو دون تغيير .

4 – غير مسموح بوجود المستوى المنطقي 1 على المدخلين S,R في نفس الوقت نظرا لانه يمثل الحالة الفعالة للبوابة NOR , ومن ثم تصبح المخارج في هذه الحالة غير معرفة كما في السطر الاخير من الجدول .

5 – حالة المخارج تتغير فقط عندما تتغير المداخل وتحتفظ المخارج بحالتها بدون اي تغير اذا بقيت المداخل بدون تغيير , اي ان دائرة المساك تمسك على حالة معينة اذا لم تتغير المداخل , ولها خاصية الاحتفاظ بالبيانات بصفة مؤقتة .

ويمكن بناء دائرة المساك من بوابتي NAND كما موضحة في الشكل التالي ونظرا لان المستوى الفعال لبوابة NAND هو 0 فان جدول الحقيقة يكون كما هو موضح في الجدول التالي , وتسمى الدائرة في هذه الحالة بدائرة المساك ذات المدخلات الفعالة المنخفضة (Active Low Inputs) :




دائرة المساك S-R ذو المدخلات الفعالة المنخفضة

| المدخلات | | الخرج | وضع التشغيل |
| --- | --- | --- | --- |
| $\overline{S}$ | $\overline{R}$ | Q | (Mode of Operation) |
| ٠ | ٠ | ? | وضع الخطر أو وضع غير مسموح به<br>Invalid condition |
| ٠ | ١ | ١ | الوضع الفعال<br>Latch SETS |
| ١ | ٠ | ٠ | الوضع غير الفعال<br>Latch RESETS |
| ١ | ١ | Q. | وضع الإمساك (عدم التغير)<br>No Change |

جدول الحقيقة لدائرة المساك S-R ذات المدخلات المنخفضة

وبالنظر الى جدول الحقيقة اعلاه يمكننا ملاحظة ما يأتي :

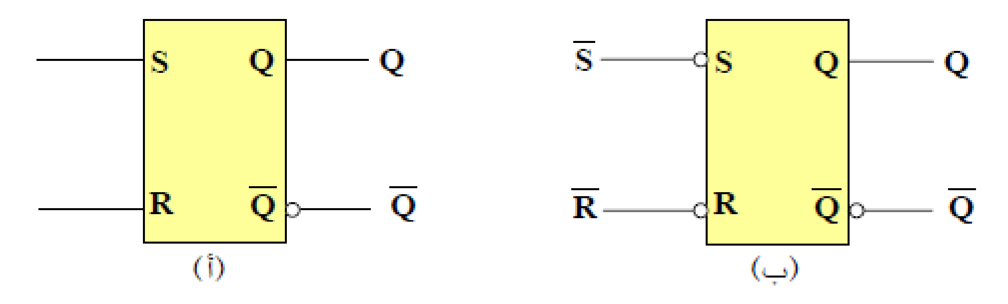
1 – وجود المستوى المنطقي 1 على المدخلين في نفس الوقت لا يغير حالة دائرة المساك ويظل المخرج Q كما هو ( في السطر الاخير ).

2 – عندما يكون المستوى المنطقي على المدخل R-=1 , S-=0 يتغير المستوى المنطقي للخرج الى 1 كما في السطر الثاني من الجدول اما اذا كان الخرج Q اصلا 1 فيبقى كما هو دون تغيير .

3 – عندما يكون المتسوى المنطقي على مدخل R-=0 , S-=1 يتغير المستوى المنطقي للخرج الى 0 كما في السطر الثالث من الجدول اما اذا كان الخرج Q اصلا 0 فيبقى كما هو دون تغيير .

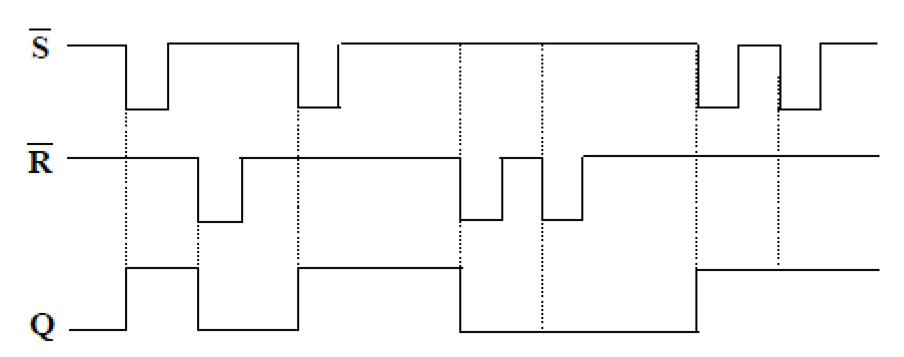
4 – غير مسموح بوجود المستوى المنطقي 0 على المدخلين بنفس الوقت نظرا لانه لا يمثل المستوى الفعال لبوابة NAND ومن ثم فان حالة المخارج تكون غير معروفة.

الشكل التالي يوضح الرمز المنطقي (Logic Symbol) لدائرة المساك ذات المدخلات الفعالة العالية ودائرة المساك ذو المدخلات الفعالة المنخفضة .



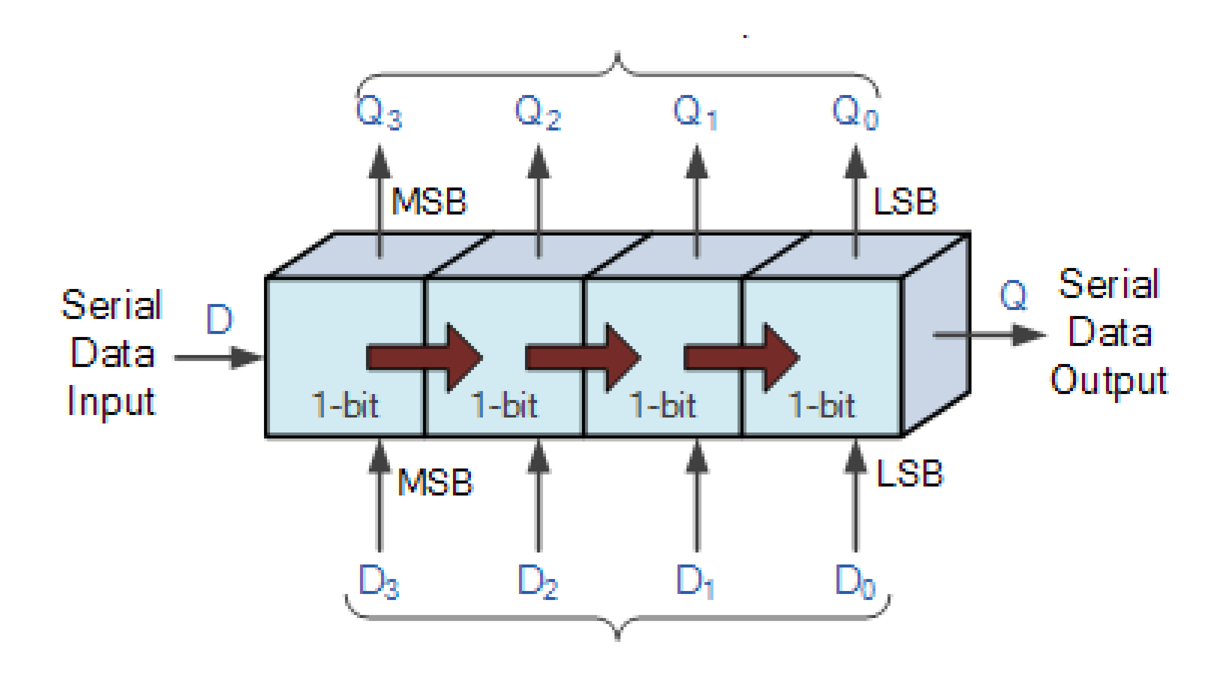

المثال التالي يوضح كيفية عمل دائرة المساك ذات المدخلات الفعالة المنخفضة وذلك عن طريق وضع نبضات على كل من -R , -S وملاحظة شكل المخرج Q وسوف نتنجب وضع R-=0 , S-=0 حيث ان حالة الخرج لا تكون معروفة في هذه الحالة.

اذا كان شكل نبضات الدخل لكل من -R , -S كما في الشكل التالي سوف نرسم شكل نبضات الخرج Q بفرض ان الحالة التي عليها الخرج Q قبل تطبيق او نبضة لكللا الدخلين هي Q=0 ستكون النتيجة كالتالي :




المخطط الزمني لدائرة المساك

# مسجلات الإزاحة (Shift Registers)



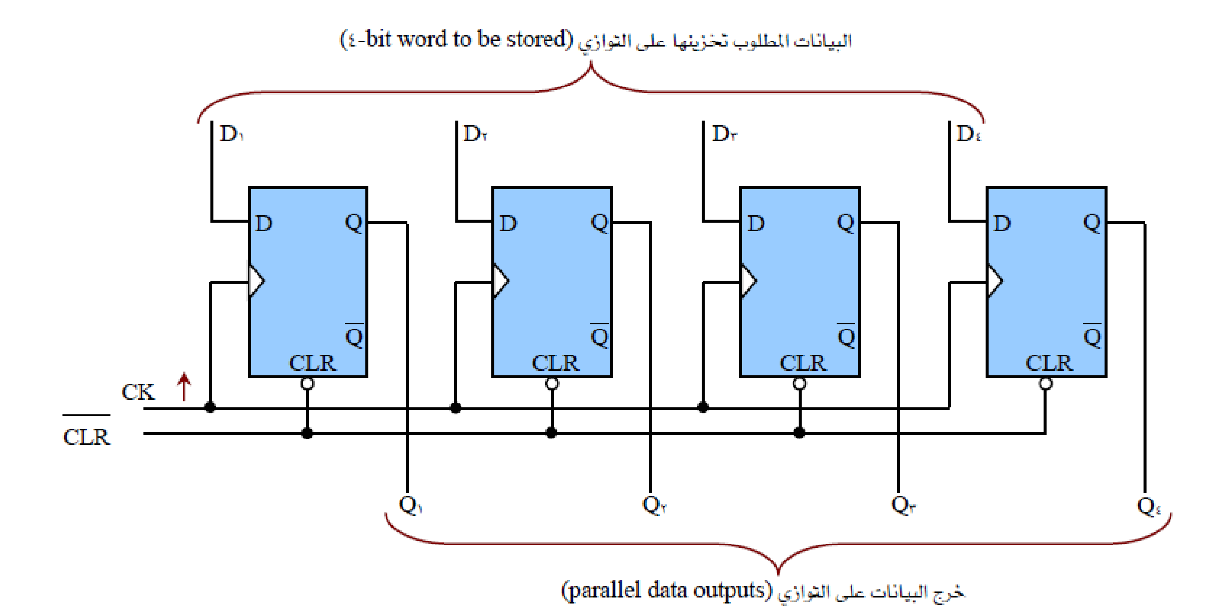

هل سبق أن وجدت نفسك بحاجة للتحكم في عدة مصابيح مُضيئة (LED’s) معاً؟ هل سبق أن وجدت نفسك بحاجة للمزيد من أطراف الإدخال والإخراج(I/O) الرقمية ؟ حسناً، في هذا الدرس سنتحدث عن تقنية ضرورية ستحتاجها في حال أردت القيام بما سبق، ويُطلق عليها مُسجلات الإزاحة.

إذا كان هناك أحد المشاريع بحاجة للتحكم بـ 16 مصباح مضيء فسيطلب ذلك استخدام 16 منفذ من منافذ الدخل/الخرج الخاصة بالمتحكم الدقيق او الدائرة المنطقية . في حال لم يكن لديك 16 منفذ مُتاح فهنا تأتي أهمية مُسجلات الإزاحة. فباستخدام اثنين من مُسجلات الإزاحة متصلين على التوالي يُمكننا التحكم بالديودات المضيئة الـ 16 من خلال أربعة منافذ دخل/خرج فقط. هذا يُشكل فرق كبير. ويمكن توفير المزيد من المنافذ عن طريق توصيل المزيد من مُسجلات الإزاحة معاً.

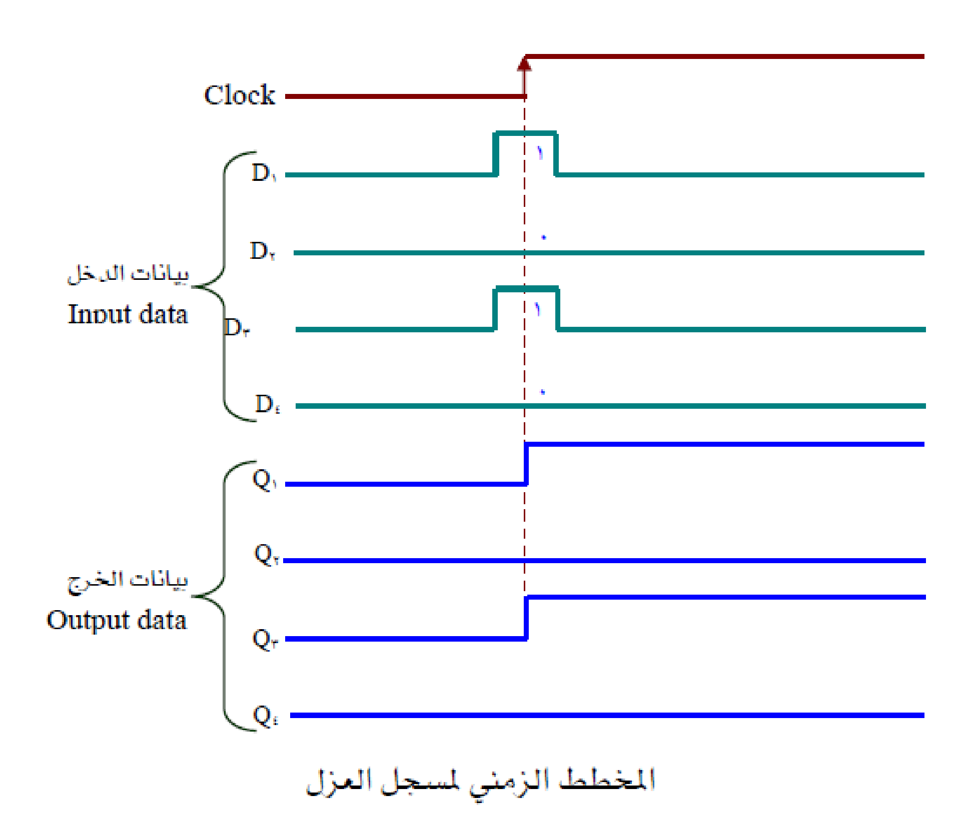
تعتبر مسجلات الازاحة احد انواع الدوائر المنطقية التعاقبية وتستخدم المسجلات عادة لتخزين البيانات , ومن دراستنا السابقة للدوائر القلابة وجدنا انه يمكن تخزين رقم ثنائي مفرد (bit) بواسطة دائرة قلابة مفردة , ومن ثم يمكن توصيل عدد من الدوائر القلابة معا لبناء ما يعرف بالمسجل والذي يستخدم كذاكرة تخرين مؤقتة اي كمية صغيرة من البيانات ولفترة زمنية قصيرة وذلك تمهيدا لنقلها كما في مسجلات النقل او العزل (Buffer Register) او لازاحة البيانات الى اليسار (Shift Left) او اليمين (Right Shift) او تحويل البيانات المتوالية (Serial Data) الى بيانات متوازية (Parallel Data) والعكس كما في مسجلات الازاحة (Shift Registers) .

اولا : مسجلات العزل (Buffer Registers)

مسجل العزل ببساطة يستخدم لتخزين كلمة رقمية (Digital word) مكونة من مجموعة من الارقام الثنائية (bits) والشكل التالي يوضح كيفية بناء مسجل عزل مكون من اربعة مراحل (4 Stages) باستخدام دوائر القلابات نوع D والتي يتم تنشيطها عند الحافة الموجبة لنبضة التزامن (Positive edge-triggered) :




مسجل عزل مكون من اربع مراحل باستخدام دوائر القلابات من النوع D



المخطط الزمني لمسجل العزل

البيانات المطلوب تخزينها والتي تتكون من اربعة ارقام ثنائية (4 - bits) تطبق على المداخل D1 , D2 , D3 , D4 للمسجل وتظهر على المخارج Q1 , Q2 , Q3 Q4 عند حدوث اول نبضة تزامن موجبة عند مدخل نبضات التزامن CK .

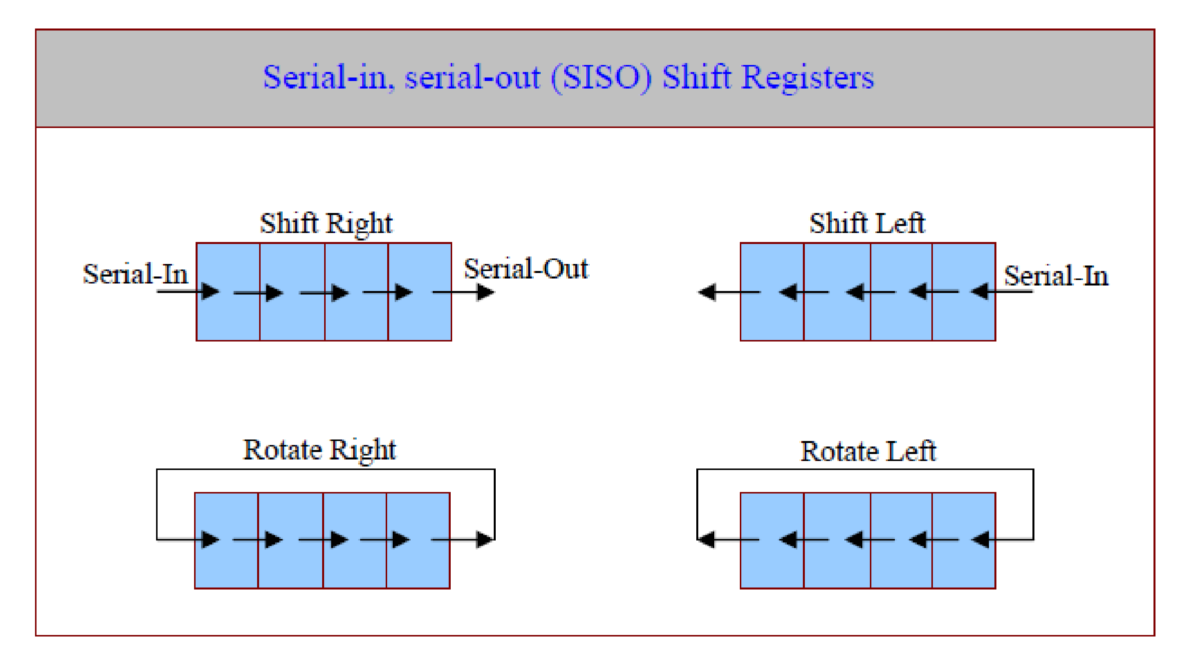
وبالرجوع الى الرسم البياني السابق ( المخطط الزمني لمسجل العزل ) نرى ان البيانات المراد تخزينها والتي تكون موجودة على خطوط البيانات Q1 , Q2 , Q3 , Q4 يتم تخزينها او ادخالها في المسجل عند الحافة الموجبة لنبضة التزامن , هذه البيانات تكون موجودة بصفة مستمرة على الخرج .

وحيث انه تم ادخال كلمة مكونة من اربعة ارقام ثنائية على التوازي لمدخل المسجل وتم اخراجها على التوازي ايضا , لذلك فان مسجلات العزل غالبا ما تسمى بمسجلات متوازية المدخل – متوازي المخرج (Parallel-in, Parallel-out Registers).
ودخل المسح (Clear-input) والمنشط عند الحافة السالبة (active-low) يستخدم لمسح جميع انواع القلابات (مسح الكلمة فقط) .

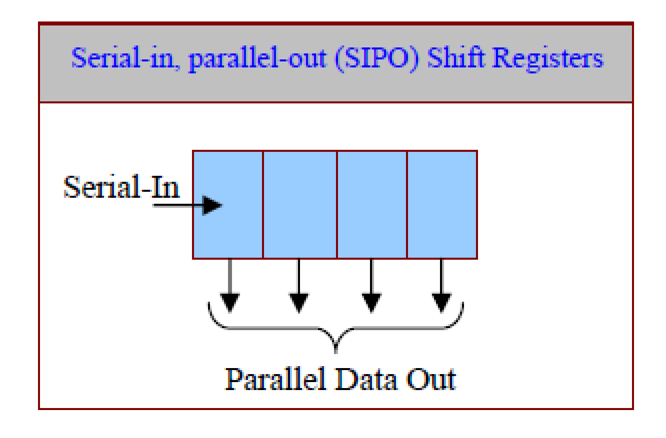
ثانيا : مسجلات الازاحة (Shift Registers)

مسجل الازاحة هو مسجل لتخزين البيانات تمهيدا لتحريكها او ازاحتها يسارا او يمينا والانواع الاساسية الثلاثة لمسجلات الازاحة هم ما يلي :

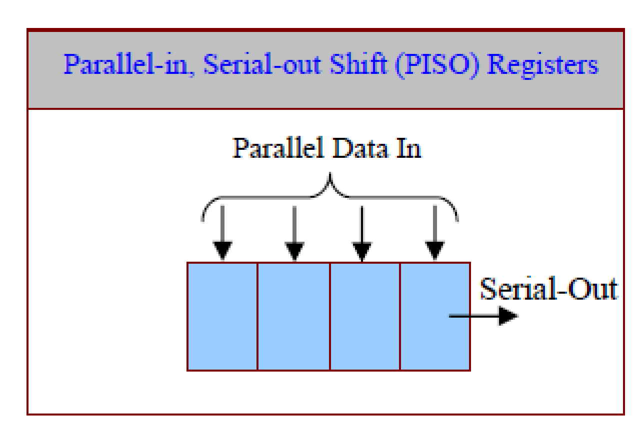
1 – مسجلات ازاحة متوالية المدخل – متوالية المخرج ( Serial-in, Serial-out Shift Registers) وتكتب اختصارا (SISO) .




2 - مسجلات ازاحة متوالية المدخل – متوازية المخرج (-Serial-in, Parallel out Shift Registers) وتكتب اختصارا (SIPO) .



3 - مسجلات ازاحة متوازية المدخل – متوالية المخرج (Parallel-in, Serial-out Shift Registers) وتكتب اختصارا (PISO) .




ولفهم كيفية تشغيل هذه المسجلات بتفصيل أكثر فلنأخذ بالتفصيل كل نوع من الانواع الثلاثة على حده :

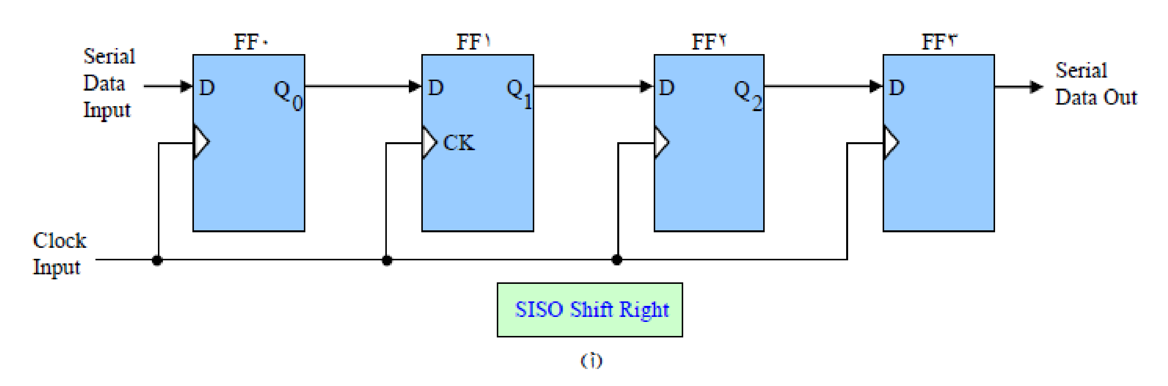
1 – مسجلات ازاحة متوالية المدخل – متوالية المخرج ( Serial-in, Serial-out Shift Registers) او (SISO) .

يوضح الجدول التالي كيفية مسجل الازاحة ففي هذا المثال نجد ان المسجل يحتوي على البيانات 0110 (محتوى ابتدائي) بينما البيانات الخارجية المتوالية 1001 موجود على دخل المسجل في انتظار حدوث ازاحة لها :

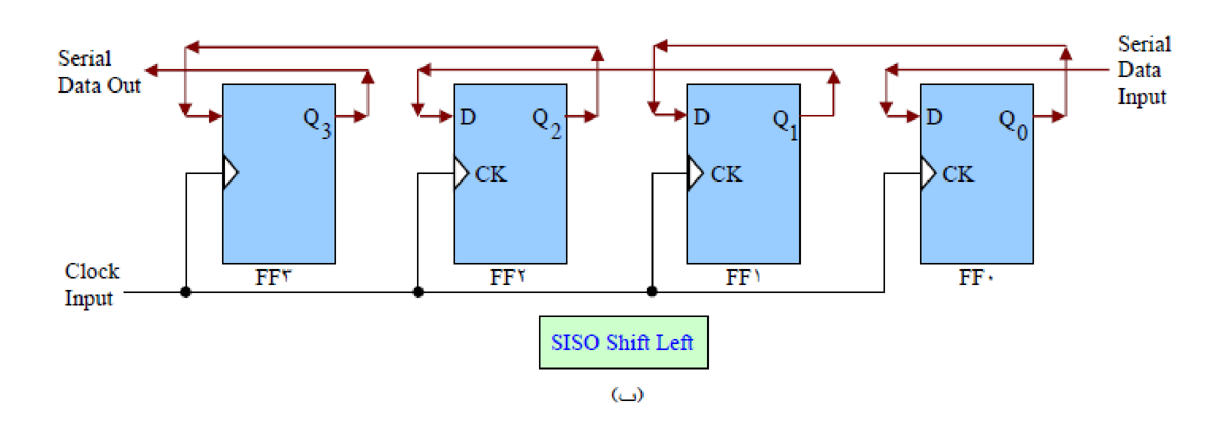
| نبضات التزامن | البيانات المراد تخزينها | خرج المسجل | | | |
|---|---|---|---|---|---|
| Clock | Input | $Q_0$ | $Q_1$ | $Q_2$ | $Q_3$ |
| — | — | ٠ | ١ | ١ | ٠ |
| ١st | ١ | ١ | ٠ | ١ | ١ |
| ٢nd | ٠ | ٠ | ١ | ٠ | ١ |
| ٣rd | ٠ | ٠ | ٠ | ١ | ٠ |
| ٤th | ١ | ١ | ٠ | ٠ | ١ |

بعد نبضة التزامن الاولى ($1^{st}$ Clock pulse) البيانات المخزنة بالمسجل سوف يحدث لها ازاحة بمقدار خانة واحدة إلى اليمين وفي نفس الوقت فان الرقم الاول من البيانات الخارجية المتوالية سوف يحدث له ازاحة داخل الخانة الاولى من المسجل . وبعد نبضة التزامن الثانية ($2^{nd}$ Clock pulse) يكون هناك رقمين من الارقام المخزنة 0110 قد تمت ازاحتها خارج المسجل بينما تم تخزين رقمين من الارقام الخارجية المتوالية 1001 . وبعد نبضة التزامن الثالثة ($3^{rd}$ Clock pulse) ثلاث ازاحات في اتجاه اليمين تكون قد تمت , وبعد نبضة التزامن الرابعة ( $4^{th}$ Clock pulse) فان البيانات الاصلية المخزنة 0110 تكون قد حدث لها ازاحة خارج المسجل بينما البيانات المطبقة على الدخل 1001 حدث لها ازاحة بالكامل داخل المسجل وهو الآن مخزنة به .

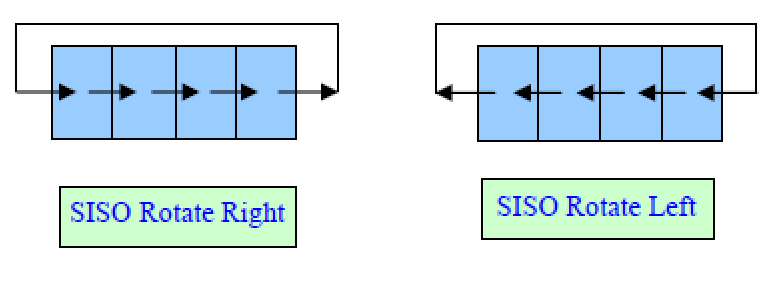
الآن نظرية التشغيل الاساسية لمسجل الازاحة تم فهمها , سوف نرى كيف يمكن استخدام دوائر القلابات لبناء دائرة مسجل الازاحة .




(أ)




(ب)




(جـ)

يوضح المخطط أ مسجل ازاحة مكون من اربع مراحل 4 bit وذلك باستخدام دائرة القلاب من النوع D . البيانات المتوالية يتم ادخالها الى الطرف D لدائرة القلاب الاولى FF0 وخرج دائرة القلاب الاول Q0 يوصل الى مدخل D لدائرة القلاب الثاني FF1 , وخرج دائرة القلاب الثاني Q1 يوصل الى دخل دائرة القلاب الثالث FF2 , ومخرج دائرة القلاب الثالث Q2 يوصل الى داخل دائرب القلاب الرابع FF3 , ومخرج دائرة القلاب الرابع يمثل الخرج المتوالي النهائي لدائرة المسجل المكون من اربع مراحل.

نبضات التزامن (Clock input) توضع لحظيا على كل دوائر القلابات , ومع كل حافة موجبة (Positive edge) كم النبضات يتم ازاحة خانة واحدة (1 bit) من بيانات الدخل الى المسجل وبالتالي فان مسجل الازاحة متوالي الدخل – متوالي الخرج يحتاج الى اربع نبضات تزامن ليتم تسجيل البيانات الاربعة الموجودة على المدخل , ومن ناحية اخرى فان هذا المسجل يحتاج الى اربعة نبضات اخرى لازاحة المعلومات الى الخارج .

وتلخيصا لما سبق شرحه , فان الدائرة (أ) في الشكل السابق تبين لنا كيفية توصيل عدد عدد اربع دوائر قلابة من النوع D وذلك لبناء مسجل ازاحة الى اليمين من النوع متوالي الدخل – متوالي الخرج (SISO Shift-Right Shift Register) .

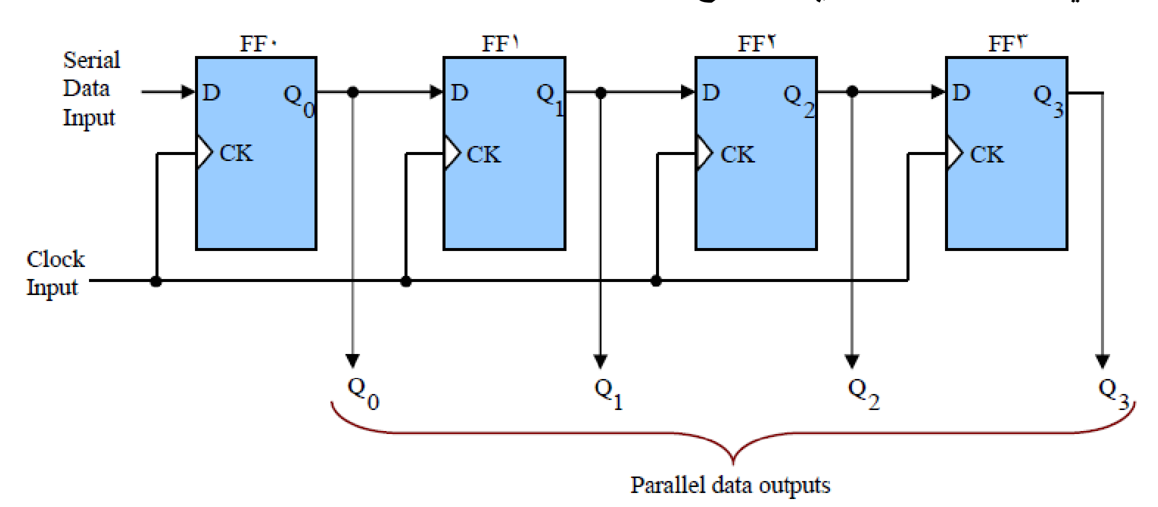
والدائرة (ب) من الشكل السابق توضح لنا كيفية بناء مسجل ازاحة الى اليسار مكون من اربع دوائر قلابة من النوع D على شكل متوالي الدخل – متوالي الخرج ( SISO Shift- Left Shift Register) .

في بعض التطبيقات , البيانات المتوالية كما في الشكل أ و ب يتم توصيلها مباشرة للخلف مرة اخرى الى طرف الدخل المتوالي للمسجل بمعنى ان البيانات الخارجة يتم

تسجيلها مرة اخرى دون ان تُفقد وتسمى هذه العملية باسم توالي المدخل – توالي المخرج دوران يمين (SISO Rotate-Right) وتوالي المدخل – توالي المخرج دوران يسار (SISO Rotate-Left) كما في الشكل السابق (ج).

2 – مسجلات ازاحة متوالية المدخل – متوازية المخرج ( Serial-in, parallel Shift registers out) او (SIPO) .

الشكل التالي يوضح النوع الثاني من مسجلات الازاحة والذي يسمى مسجل ازاحة متوالي المدخل – متوازي المخرج :



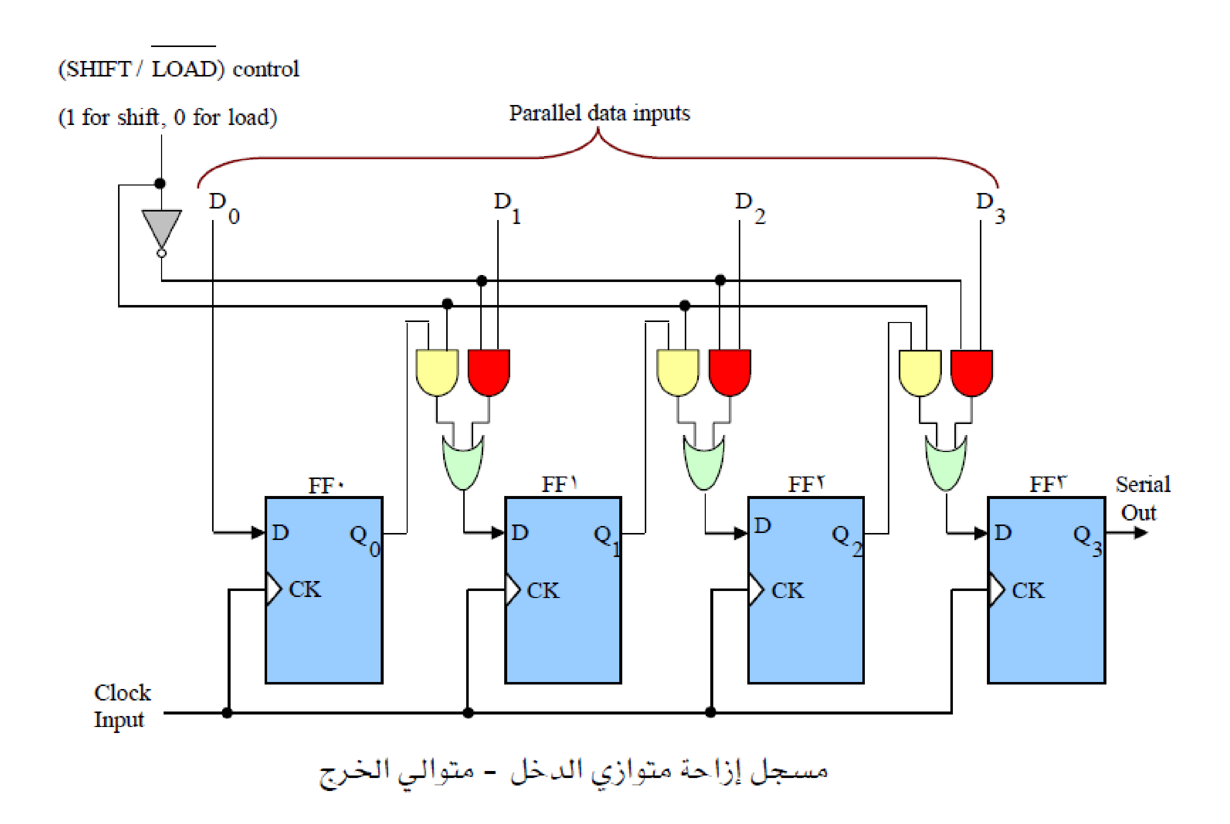

مسجل ازاحة متوالي الدخل -متوازي الخرج

لادخال البيانات في هذا المسجل يتم تطبيق البيانات المتوالية والمكونة من 4 bit على مدخل البيانات على التوالي (Serial data input) ويتم ازاحتها تحت التحكم في نبضات الدخل المتزامنة اي ( ازاحة واحدة في اتجاه اليمين لكل نبضة ).

ولادخال او تخزين كلمة مكونة من اربعة ارقام 4 bits على التوالي داخل هذا المسجل فاننا نحتاج الى اربع نبضات تزامن . البيانات المخزنة داخل مسجل الازاحة تكون موجودة على المخارج الاربعة ($Q_0$,Q1,Q2,Q3) كاربعة ارقام 4 bits خرج على التوازي .

3 – مسجلات ازاحة متوازية المدخل – متوالية المخرج (Parallel-in, Serial-Shift registers out) او (PISO) .

يوضح الشكل التالي كيفية بناء مسجل مكون من اربع مراحل من النوع متوازي الدخل – متوالي الخرج وذلك باستخدام دوائر القلابات من النوع D .



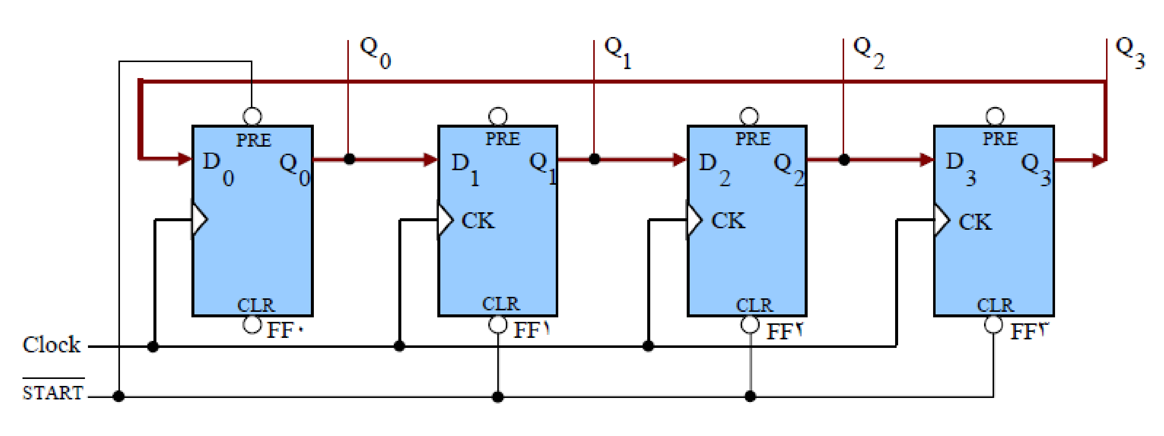
مسجل إزاحة متوازي الدخل - متوالي الخرج

يتم التحكم في الدائرة عن طريق طرف التحكم الداخل SHIFT/LOAD , عندما يكون طرف التحكم SHIFT/LOAD في الوضع LOW , فان جميع البوابات AND المظللة باللون الاحمر تكون نشيطة (Enabled) نتيجة لعكس اشارة التحكم عن طريق العاكس (Inverter) المظلل . هذه البوابات الفعالة تعمل على توصيل البيانات من خطوط الدخل للبيانات $(D_0$ , D1 , D2 , D3) الى مداخل البيانات على دوائر القلابات . وعند وصول نبضة التزامن (Clock pulse) فان هذه البينات سوف يتم تخزينها داخل المسجل وتظهر على المخارج (Q0,Q1,Q2,Q3).

وعندما يكون طرف التحكم SHIFT/LOAD في الوضع High فان جميع بوابات AND المظللة باللون الاصفر تكون فعالة او نشطة , هذه البوابات الفعالة توصل الرج $Q_0$ الى الدخل D لدائرة القلاب الثاني FF1 , وتوصل الخرج Q1 الى دخل دائرة القلاب الثالث FF2 , وكذلك توصيل الخرج Q1 Q2 الى دخل دائرة القلاب الرابع FF3 . وفي هذا الوضع فان البيانات المخزنة داخل مسجل الازاحة سوف تحدث لها ازاحة من جهة اليمين وبمقدار خانة واحدة 1 bit مع كل نبضة من نبضات التزامن الموجودة على الدخل (clock input) .

4 – مسجل الازاحة المتتابع (عداد حلقي) Shift Register (Ring Counter) Sequencer

يوضح الشكل التالي كيفية توصيل مسجل الازاحة على شكل عداد حلقي وذلك بتوصيل خرج الدائرة القلابة FF3 الى دخل الدائرة القلابة FF0 ( توصيل الخرج Q3 بالدخل D) :



كيفية توصيل مسجل الإزاحة على شكل عداد حلقي

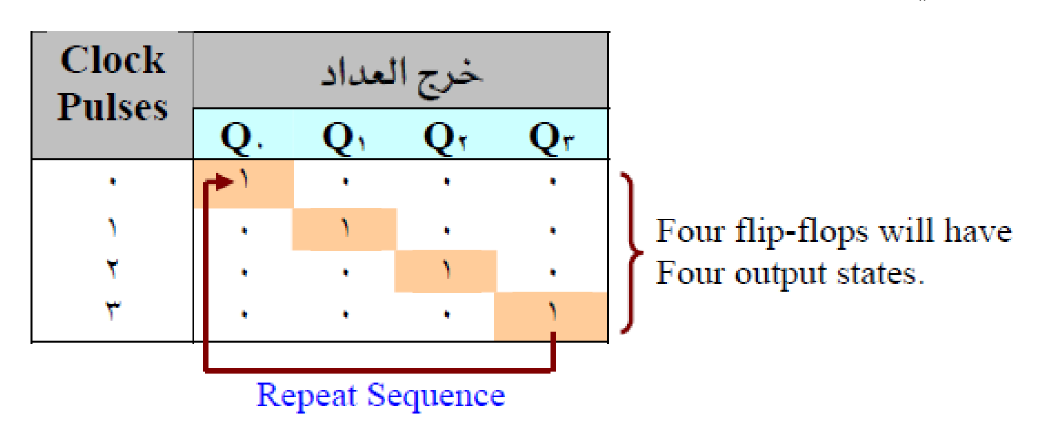
ان هذه الخاصية الدائرية او الحلقية تجعل انتقال البيانات داخل مسجل الازاحة في شكل دائري او حلقي , فعندما يكون خط التحكم SRART في المستوى LOW فان الخرج $Q_0$ سوف يصبح في المستوى High (PRE = 0) والمخارج Q1 , Q2 , Q3 في

المستوى LOW (CLR = 0) كما هو موضح في الرسم البياني التالي والذي يمثل النبضات :

Clock
$\overline{START}$
$Q_0$
$Q_1$
$Q_2$
$Q_3$

نبضات الخرج للعداد الحلقي

والشكل التالي يمثل جدول الحقيقة لهذا العداد :



| Clock Pulses | خرج العداد | | | |
|---|---|---|---|---|
| | Q٠ | Q١ | Q٢ | Q٣ |
| ٠ | ١ | ٠ | ٠ | ٠ |
| ١ | ٠ | ١ | ٠ | ٠ |
| ٢ | ٠ | ٠ | ١ | ٠ |
| ٣ | ٠ | ٠ | ٠ | ١ |

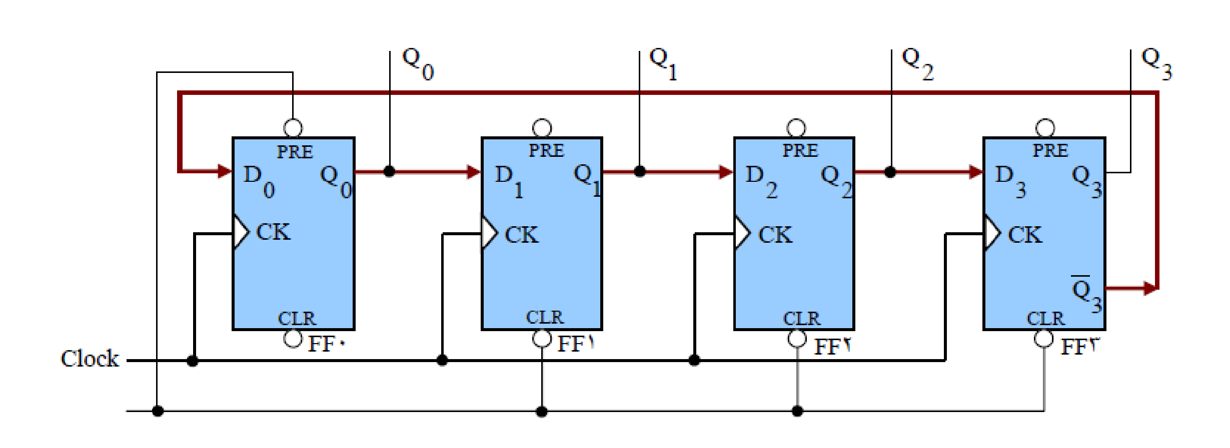


الكلمة المسجلة الآن اصبحت (1000) سوف يحدث لها ازاحة من جهة اليمين مع كل نبضات التزامن

والرقم 1 الموجود في الكلمة المسجلة سوف يزاح بشكل دائري داخل المسجل كما هو موضح في جدول الحقيقة اعلاه .

5 – عداد جونسون Johnson Counter

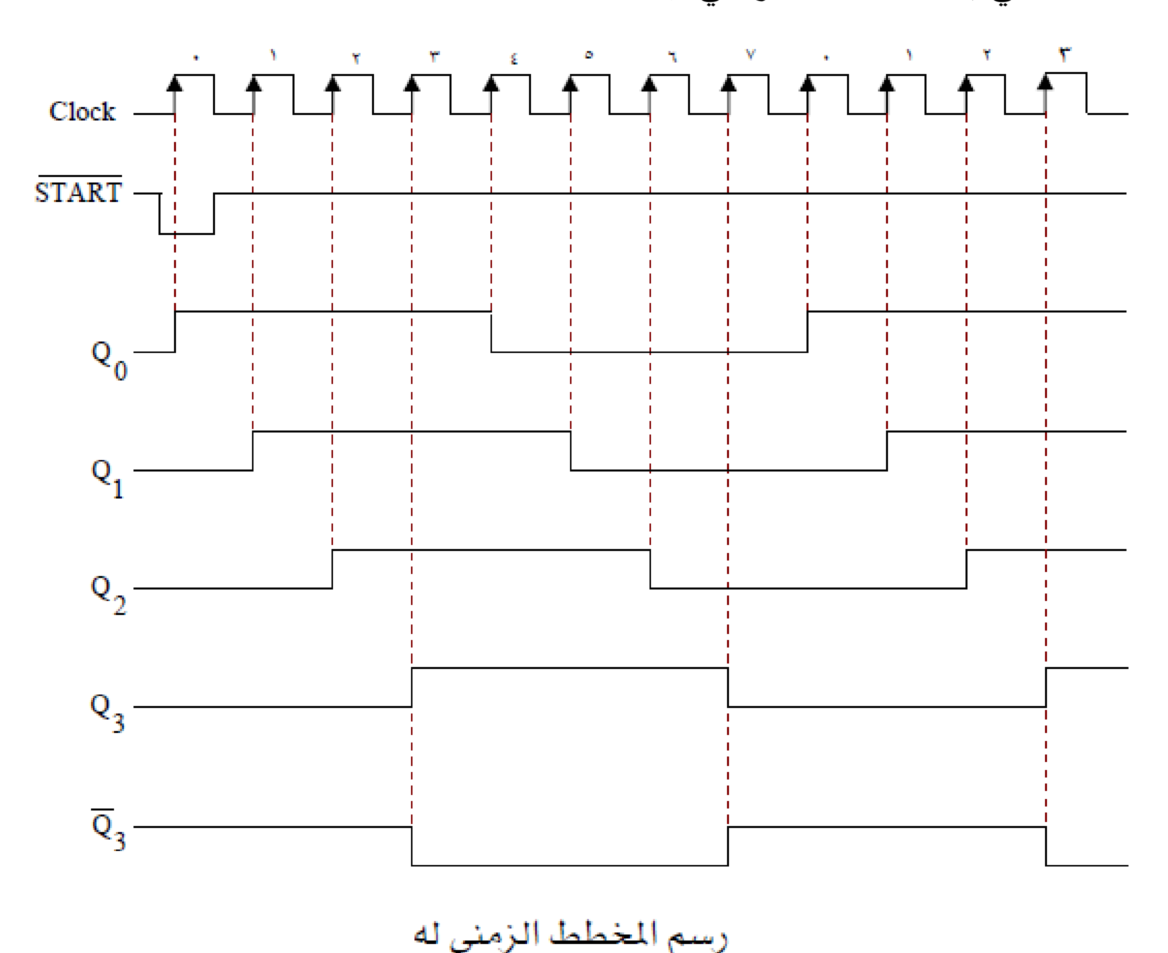
يبين الشكل التالي دائرة مسجل ازاحة موصلة على هيئة عداد جونسون , وكما نرى ان عداد جونسون يتم بناؤه تماما بنفس طريقة العداد الحلقي فيما عدا ان الخرج المعكوس لآخر دائرة قلابة Q3 هو الذي يوصل بدخل الدائرة القلابة D0 :




ومثل العداد الحلقي , فإن عداد جونسون يحتاج الى تجهيز الخرج الابتدائي للدائرة كما نرى من شكل النبضات في المخطط الزمن لهذا المسجل وجدول الحقيقة الخاص به وهو (1000) وبما ان Q3 في المستوى LOW عند البداية فان $Q3^-$ سوف تكون في المستوى HIGH وهذا المستوى سوف يعاد تغذيته الى الدخل $D_0$ , وبالتالي فان المداخل ذات المستويات العالية يتم ادخالها الى داخل مسجل الازاحة من اليسار الى اليمين الى ان يصبح خرج جميع دوائر القلابات يساوي HIGH , وعندما تصبح Q3 عند المستوى HIGH ( بعد نبضة التزامن الثالثة ) , $Q3^-$ سوف يكون عند المستوى LOW وبالتالي فان $D_0$ تصبح ايضا LOW . مسجل الازاحة الآن سوف يبدأ في عمل ازاحة لهذه المستويات المنخفضة من اليسار الى اليمين الى ان يصبح خرج جميع دوائر القلابات يساوي LOW وعندما تصبح Q3 عند المستوى LOW ( بعد نبضة

التزامن السابقة ) , $Q3^{-}$ سوف يكون عند المستوى HIGH وبالتالي فان $D_0$ تصبح ايضا HIGH مما يتسبب في تكرار دورة الازاحة مرة اخرى و هكذا .

الشكل التالي يمثل المخطط الزمني لهذا العداد :




رسم المخطط الزمني له

في العداد الحلقي يكون عدد حالات الخرج المختلفة محكومة بعدد الدوائر القلابة في المسجل , وبناءا عليه فان العداد الحلقي المكون من اربعة مراحل سوف يعطي اربع حالات حالات مختلفة للخرج ( كما في جدول الحقيقة )

والشكل التالي يبين جدول الحقيقة لهذا العداد :

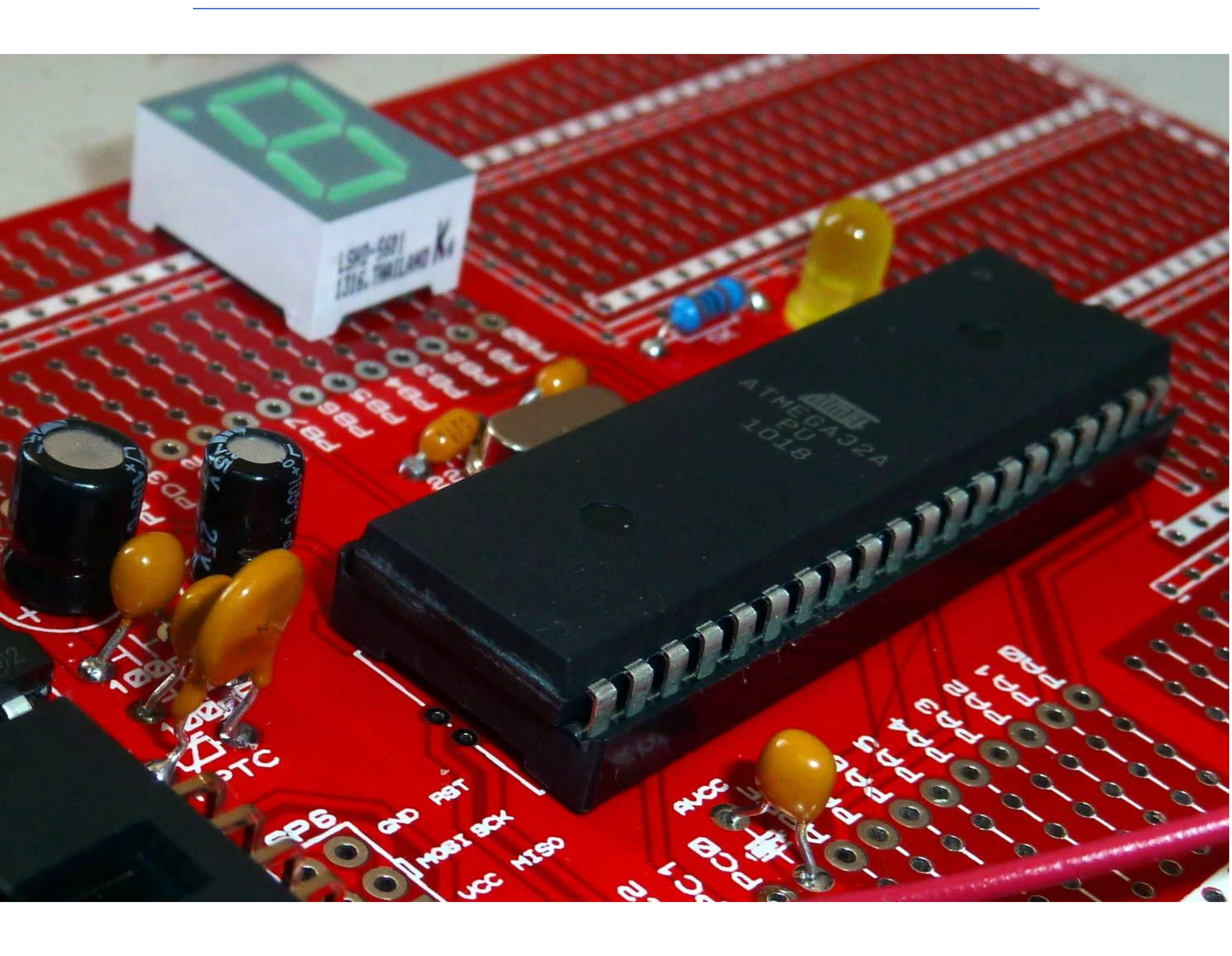
| Clock Pulses | خرج العداد | | | | |
|---|---|---|---|---|---|
| | $Q_0$ | $Q_1$ | $Q_2$ | $Q_3$ | $\overline{Q_3}$ |
| ٠ | ١ | ٠ | ٠ | ٠ | ١ |
| ١ | ١ | ١ | ٠ | ٠ | ١ |
| ٢ | ١ | ١ | ١ | ٠ | ١ |
| ٣ | ١ | ١ | ١ | ١ | ٠ |
| ٤ | ٠ | ١ | ١ | ١ | ٠ |
| ٥ | ٠ | ٠ | ١ | ١ | ٠ |
| ٦ | ٠ | ٠ | ٠ | ١ | ٠ |
| ٧ | ٠ | ٠ | ٠ | ٠ | ١ |

Four flip-flops will have eight output states.

Repeat Sequence

وفي عداد جونسون يكون عدد حالات الخرج المختلفة يساوي ضعف عدد الدوائر القلابة في المسجل ففي هذه الدائرة يكون لدينا ثماني حالات مختلفة للخرج (2 × 4 flip-flops = 8) كما في جدول الحقيقة .

## الميكروكونترولر




هو عبارة عن قطعة الكترونية رقمية صغيرة تم اختراعها بعد الكمبيوترات التى تقوم بتخزين البرامج ويقوم الميكروكنترولر بحفظ مجموعة من التعليمات بداخله والتي تسمى برنامج والتى يكون من السهل التعديل فيها بدلا من اعادة تغيير الاسلاك والتوصيلات كما كان متبع قديما في الدوائر المتكاملة IC.

تعرفنا سابقا على المتحكم الالكتروني القابل للبرمجة الميكرو كونترولر وقمنا بشرح مكوناته , في هذه الدروس سوف نبدأ التعامل معه والتعرف على انواعه المختلفة وكيفية ربطه في قطعة البرمجة المسؤولة عن ربطه بجهاز الحاسوب , وسوف نتعرف ايضا على المذبذب (الكريستالة) وابرز لغات برمجته وايضا سوف نقوم بعمل مشاريع عملية

وابرز المكونات الداخلية للميكرو كونترولر :

- وحدة المعالجة المركزية Central Processing Unit CPU
- ذاكرة مؤقتة (عشوائية)Rondom Access Memory RAM
- ذاكرة قابلة للقراءة و الكتابة EEPROM
- مداخل الربط الرقمية والتناظرية I/O Units
- عدادات ومولدات اشارة وبعض المكونات الاختيارية حسب الطراز/الموديل

اما عن استخداماته يوجد الكثير من الأشياء التي يمكنك القيام بها باستخدام متحكم دقيق

مثلا يمكن بناء الروبوت أو مشغل MP3 أو الهاتف المحمول أو قفل الباب الذي يفتح الباب تلقائيًا عند إدخال رمز على هاتفك الذكي يمكننا القول ان الاحتمالات لا حصر لها!

لنفترض أنك تريد إنشاء إنسان آلي-روبوت على شكل انسان , يمكنك توصيل مستشعر الأشعة تحت الحمراء لاستخدامه كرؤية للروبوت , ويمكنك توصيل محرك ببعض العجلات لجعله يتحرك.

الآن ، كل ما عليك فعله هو عمل برنامج يقرأ من مستشعر الأشعة تحت الحمراء ويتحكم في المحرك. في شفرتك ، يمكنك التأكد من توقف الروبوت إذا رأى شيئًا أمامه واتجه إلى اليسار أو اليمين قبل المتابعة.

عندما تعرف كيفية بناء دوائر متحكم دقيق ، لا توجد حدود تقريبًا لما يمكنك فعله! وباتباع هذا الدرس التعليمي ستتعلم استخدام ال Microcontrollers في المشاريع الخاصة بك , و في هذه الدروس سوف نشرح الاساسيات لتبدأ رحلتك الموسعة في هذا المجال المتقدم من اقسام التكنولوجيا .

وحدة التحكم الدقيقة لديها عدة دبابيس-اطراف. معظم هذه الدبابيس (PINS) تسمى مسامير الإدخال والإخراج وباستخدام هذه المسامير ، يمكن أن يتفاعل الميكروكونترولر مع العالم الخارجي.

لا يقوم الميكروكونترولر بأي شيء بحد ذاته فتحتاج إلى معرفة ما يجب القيام به ، عن طريق إنشاء برنامج تقوم بتحميله عليه هذا ما يسمى ببرمجة الميكروكونترولر.فمن البرنامج الذي تكتبه ، يمكنك التحكم في دبابيس الإدخال والإخراج فقط عدا عن ذلك لن يفعل شيئ. لذا - من خلال توصيل شيء ما ، مثل صمام ثنائي باعث للضوء اي مصباح (LED) إلى طرف إخراج ، ستتمكن من تشغيل وإيقاف تشغيل الضوء من البرنامج او الكود .

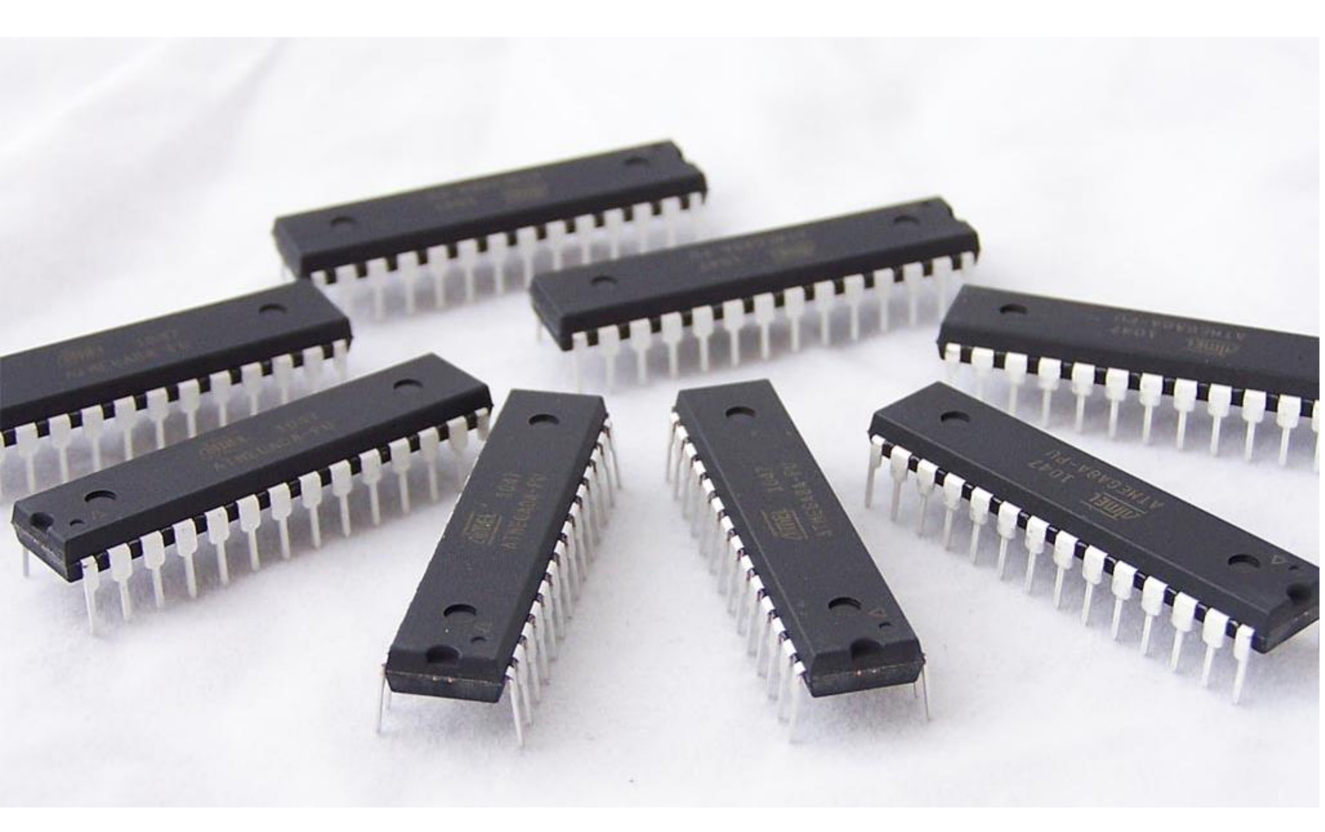
**Pin Configuration**

| | | | |
|---|---|---|---|
| PC6 | 1 | 28 | PC5 |
| PD0 | 2 | 27 | PC4 |
| PD1 | 3 | 26 | PC3 |
| PD2 | 4 | 25 | PC2 |
| PD3 | 5 | 24 | PC1 |
| PD4 | 6 | 23 | PC0 |
| VCC | 7 | 22 | GND |
| GND | 8 | 21 | AREF |
| PB6 | 9 | 20 | AVCC |
| PB7 | 10 | 19 | PB5 |
| PD5 | 11 | 18 | PB4 |
| PD6 | 12 | 17 | PB3 |
| PD7 | 13 | 16 | PB2 |
| PB0 | 14 | 15 | PB1 |

يمكن استخدام دبوس الإدخال للتحقق مما إذا تم الضغط على زر متصل به. أو لقراءة درجة الحرارة من جهاز استشعار درجة الحرارة.

وفي برنامجك ، ستتمكن من اتخاذ القرارات استنادًا إلى المدخلات ونوع المخرجات وذلك حتى تتمكن من إنشاء برنامج يبدأ بوميض الضوء إذا كانت درجة الحرارة أعلى أو أدنى من مستوى معين والى انظمة غاية في التعقيد وبعض السيارات تحتوي على مايزيد عن 50 وحدة من هذه المتحكمات . وفي هذه السلسلة سوف ندرس انواع المدخلات والمخرجات التي تمكنك من بناء كل ما ترغب به .

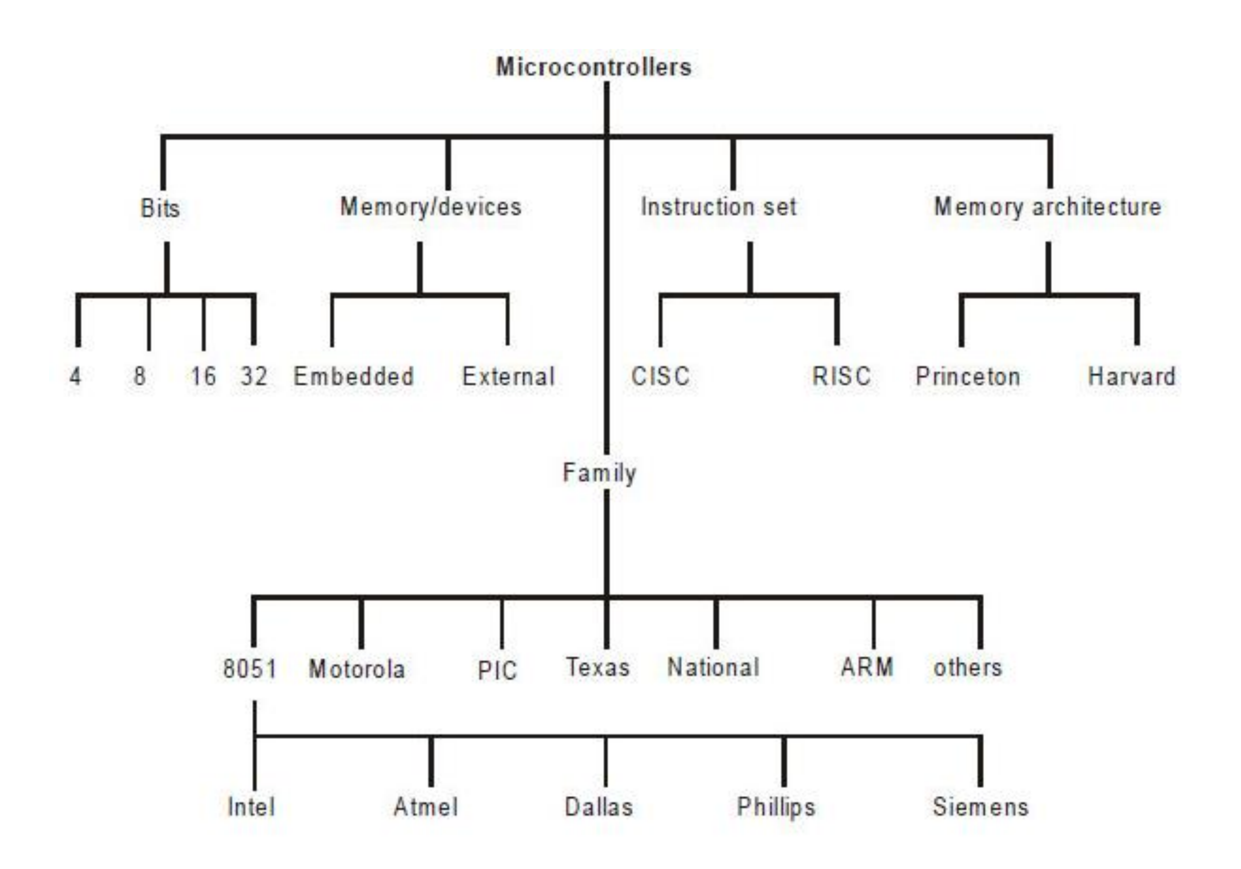
# *انواع الميكروكونترولر*




تتميز الميكروكنترولر فيما يتعلق بالمداخل ، ومجموعة التعليمات ، وبنية الذاكرة. بالنسبة للعائلة نفسها ، قد تكون هناك أشكال مختلفة بمصادر مختلفة.

يجب أن يكون لمتحكمات Microcontrollers عادة متطلبات منخفضة الطاقة لأن العديد من الأجهزة التي تتحكم فيها تعمل بالبطاريات. يتم استخدام Microcontrollers في العديد من الأجهزة الإلكترونية الاستهلاكية ومحركات السيارات وملحقات الكمبيوتر وأجهزة الاختبار أو القياس. وهذه هي مناسبة تماما لتطبيقات البطارية طويلة الأمد. سوف نوضح في هذا الدرس بعض الأنواع الأساسية للميكروكونترولر .

الشكل التالي يوضح أنواع المتحكمات الدقيقة ، وهي تتميز ببتاتها (Bits) وبنية الذاكرة والأجهزة ومجموعة التعليمات , دعونا نناقش الموضوع اكثر




- التصنيف حسب عدد ال Bits

قد تحتوي وحدات البت في وحدة التحكم الدقيقة على 8 بتات او 16 بت ومتحكم دقيق 32 بت.

8 بت : تحتوي على bus داخلي بمعدل 8 بت ثم تقوم ALU بتنفيذ العمليات الحسابية والمنطقية , مثال عليها Intel 8031/8051, PIC1x و عائلات Motorola MC68HC11

16 بت : يقوم بمزيد من الدقة والأداء بالمقارنة مع 8 بت , على سبيل المثال ، لا تستطيع وحدات التحكم الدقيقة ذات 8 بت استخدام الا 8 بتات ، مما يؤدي إلى نطاق نهائي من 0xFF – 0×00 اي (0-255) لكل دورة .

اما وحدات التحكم الدقيقة 16 بت تعرض البيانات 16 بت على نطاق

0xFFFF – 0×0000 اي (0-65535) لكل دورة , من المرجح أن القيمة الأطول للتوقيت مفيدة في بعض التطبيقات والدوائر

16 بت يمكن أن تعمل تلقائيا على رقمين بت , مثال عليها : 8051XA, PIC2x, Intel 8096 and Motorola MC68HC12 families.

32 بت : يستخدم microcontroller تعليمات 32 بت لتنفيذ العمليات الحسابية والمنطقية الكبيرة جدا , وتستخدم في الأجهزة التي يتم التحكم فيها تلقائيًا بما في ذلك الأجهزة الطبية القابلة للزرع وأنظمة التحكم في المحركات والآلات المكتبية والأجهزة وغيرها من الأنظمة المضمنة (Embedded Systems) والذكاء الاصطناعي .

- التصنيف حسب أجهزة الذاكرة (Memory Devices)

وتنقسم أجهزة الذاكرة إلى نوعين :

- Embedded memory microcontroller
- External memory microcontroller

حيث ان (Embedded) عندما تحتوي الميكروكونترولر على جميع الكتل الوظيفية (مكونات الميكرو كونترولر)المتاحة على الشريحة ، يدعى متحكم مدمج مثال على ذلك 8051 تحتوي على ذاكرة البرنامج والبيانات ، منافذ الإدخال / الإخراج ، الاتصال التسلسلي ، العدادات (counters) وأجهزة ضبط الوقت والتعليمات على الشريحة.

ملاحظة يهمنا في هذا النوع من الدراسات الالمام في اللغة الانجليزية !

المكونات التي ذكرناها باللغة العربية هي نفسها :

program & data memory, I/O ports, serial communication, counters and timers and interrupts .

اما النوع الثاني (External) عندما تحتوي الميكروكونترولر على كتل وظيفية ناقصة من الشريحة ، يدعى وحدة التحكم في الذاكرة الخارجية , على سبيل المثال ، 8031 ليس لديه ذاكرة برنامج على الشريحة هو متحكم ذاكرة خارجي.

- التصنيف وفقا لمجموعة التعليمات (Instruction Set)

CISC: (Complex Instruction Set Computer) تسمح للمبرمج استخدام تعليمة واحدة بدلاً من العديد من التعليمات البسيطة.

RISC: (Reduced Instruction set Computer) هذا النوع من مجموعات التعليمات يقلل من تصميم المعالج الدقيق لمعايير الصناعة , يسمح لكل تعليمة بالعمل على أي سجل أو استخدام أي وضع عنونة والوصول المتزامن للبرنامج والبيانات.

الجدول التالي يوضح امثلة على النوعين

| | | | | |
|---|---|---|---|---|
| **CISC:** | Mov AX, 4 | **RISC:** | | Mov AX, 0 |
| | Mov BX, 2 | | | Mov BX, 4 |
| | ADD BX, AX | | | Mov CX, 2 |
| | | | Begin | ADD AX, BX |
| | | | Loop | Begin |

من الجدول أعلاه ، تقتصر أنظمة RISC وقت التنفيذ من خلال تقليل دورات الساعة لكل التعليمات وأنظمة CISC تقصر وقت التنفيذ عن طريق تقليل عدد التعليمات لكل برنامج . يعطي RISC تنفيذ أفضل منCISC

- التصنيف وفقا لمعمارية الذاكرة ( Memory Architecture )

يوجد نوعان :

- Harvard memory architecture microcontroller

عندما تكون وحدة وحدة التحكم الدقيقة تحتوي على مساحة عنوان ذاكرة متباينة للذاكرة والمعلومات ، فإن وحدة التحكم الدقيقة لديها بنية ذاكرة Harvard في المعالج.

- Princeton memory architecture microcontroller

عندما يكون الميكروكونترولر يحتوي على عنوان ذاكرة مشترك لذاكرة البرنامج وذاكرة البيانات ، فإن الميكروكونترولر له بنية ذاكرة برينستون في المعالج.

- التصنيف حسب التطبيقات (Applications of Microcontrollers)

يمكن للمتحكم ان يكون لديه العديد من التطبيقات للمعدات الإلكترونية على سبيل المثال:

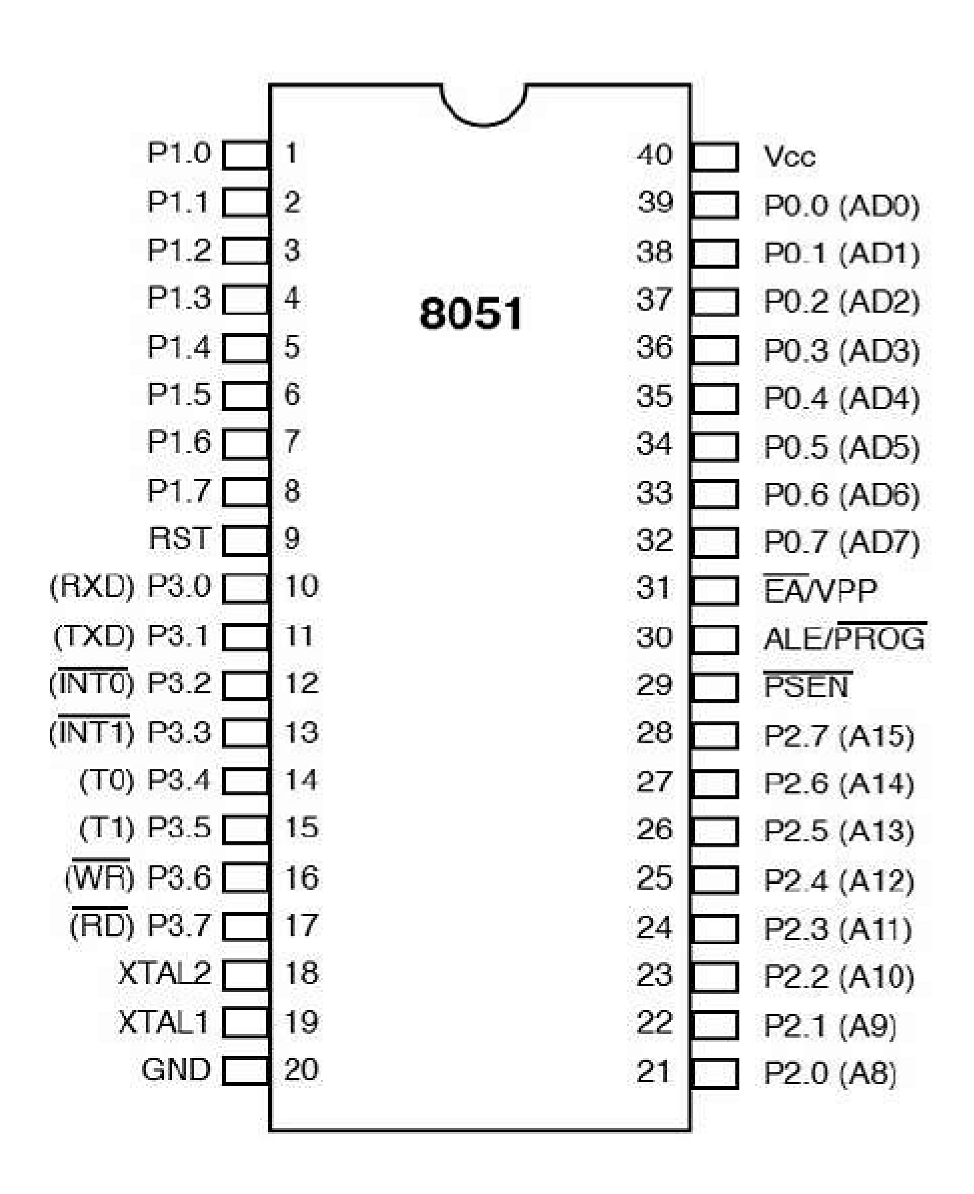
الكاميرات
الهواتف المحمولة
إنذارات الأمن
السيارات
غسالة ملابس

الآن سوف نقوم بدراسة بعض انواع المتحكمات الالكترونية وهم ما يلي :

AVR Microcontrollers
Microcontroller 8051
ATmega328
Renesas Microcontroller

- Microcontroller 8051

اهم خصائص هذا المتحكم الدقيق انه مكون من 40 دبوس مع Vcc بجهد 5 V متصلة في طرف رقم 40 , و Vss في الطرف 20 وتحفظ بقيمة 0 , V وهناك منافذ الإدخال والإخراج من P1.0 - P1.7 و يحتوي Port3 على ميزات إضافية لديه 36 طرف حالة استنزاف مفتوحة و 17 دبوس داخليا قد سحب الترانزستور داخل المتحكم. عندما نطبق المنطق 1 فيport1 ، نحصل على المنطق 1 في port21 والعكس.



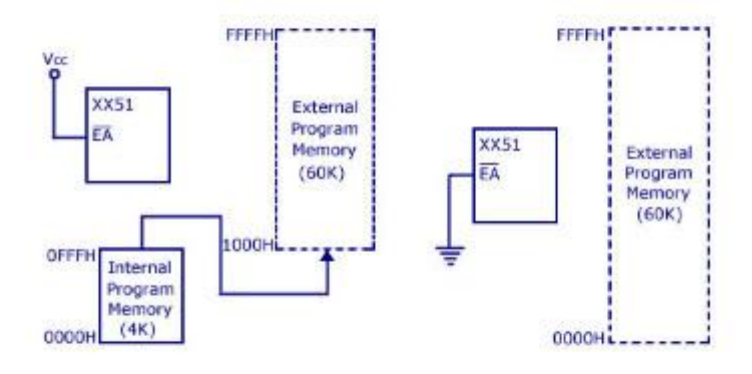
نكتب في الأساس برنامجًا في لغة C والذي يتم تحويله بعد ذلك إلى لغة الآلة التي يفهمها المتحكم الدقيق. تم توصيل دبوس RESET بـ pin9 متصلا مع مكثف.

عندما يكون المفتاح في وضع التشغيل ، يبدأ المكثف بالشحن ويكون RST عاليًا يؤدي تطبيق قيمة عالية إلى دبوس إعادة التعيين إلى إعادة تعيين وحدة التحكم الدقيقة. إذا قمنا بتطبيق منطق الصفر على هذا الدبوس ، يبدأ البرنامج في التنفيذ من البداية.

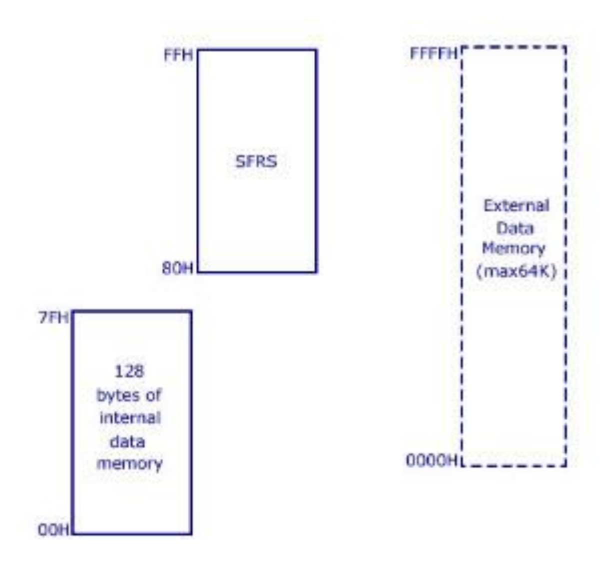
## معمارية ذاكرة 8051

تنقسم ذاكرة 8051 إلى جزئين , هما ذاكرة البرنامج وذاكرة البيانات بحيث تقوم ذاكرة البرنامج بتخزين البرنامج الجاري تنفيذه بينما تقوم ذاكرة البيانات مؤقتًا بتخزين البيانات والنتائج ويتم استخدام 8051 في عدد كبير من الأجهزة ، ويرجع ذلك أساسًا إلى سهولة دمجها في اي جهاز. وتستخدم وحدات التحكم الدقيقة هذه في إدارة الطاقة والشاشة التي تعمل باللمس والسيارات والأجهزة الطبية.

## برنامج ذاكرة 8051



## ذاكرة البيانات من 8051




## وظائف اطراف متحكم 8051 من ال Data sheet

| الاطراف | الوظيفة |
|---|---|
| Pin-40 | Vcc is the main power source of +5V DC |
| Pin 20 | Vss – it represents ground (0 V) connection |
| Pins 32-39 | Known as Port 0 (P0.0 to P0.7) to serving as I/O ports |
| Pin-31 | Address Latch Enable (ALE) is used to demultiplex the address-data signal of port 0 |
| Pin-30 | (EA) External Access input is used to enable or disable external memory interfacing. If there is no external memory requirement, this pin is always held high. |
| Pin- 29 | Program Store Enable (PSEN) is used to read signal from external program memory |
| Pins- 21-28 | Known as Port 2 (P 2.0 to P 2.7) – in addition to serving as I/O port, higher order address bus signals are multiplexed with this quasi bi directional port. |
| Pins 18 and 19 | Used to interfacing an external crystal to provide system clock. |

| | |
|---|---|
| Pins 10 – 17 | This port also serves some other functions like interrupts, timer input, control signals for external memory interfacing Read and Write. This is a quasi bidirectional port with internal pull up. |
| Pin 9 | It is a RESET pin, used to set the 8051 microcontroller to its initial values, while the microcontroller is working or at the initial start of application. The RESET pin must be set high for 2 machine cycles. |
| Pins 1 – 8 | This port does not serve any other functions. Port 1 is a quasi bi directional I/O port. |

- Renesas Microcontroller

رينيساس هي أحدث عائلة متحكم في السيارات التي توفر ميزة عالية الأداء مع استهلاك منخفض للطاقة بشكل استثنائي على مدى واسع ومتنوع من العناصر. يوفر هذا الميكروكونترولر أمانًا وظيفيًا غنيًا وخصائص أمان مضمنة مطلوبة لتطبيقات السيارات الجديدة والمتقدمة. يدعم الهيكل الأساسي لوحدة التحكم الدقيقة وحدة المعالجة المركزية الموثوقية العالية ومتطلبات الأداء العالي.

وحدة التحكم الدقيقة Renesas التي توفر طاقة منخفضة ، وأداءً عاليًا ، وحزمًا متواضعة ، وأكبر مجموعة من أحجام الذاكرة مجتمعة مع خصائص الأجهزة الطرفية الغنية.

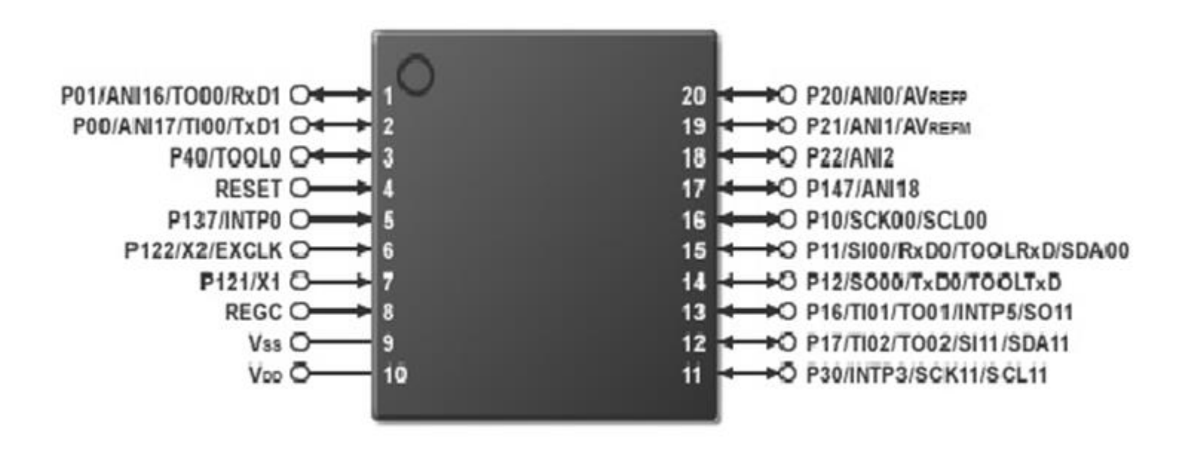


تقدم Renesas أكثر عائلات microcontroller تنوعاً في العالم على سبيل المثال تقدم عائلة RX العديد من أنواع الأجهزة مع متغيرات الذاكرة من ذاكرة 32K flash/4K RAMإلى ذاكرة 8M flash/512K RAM مذهلة !

عائلة RX من ميكروكنترولر 32 بت عبارة عن وحدة متعددة الأغراض ذات الأغراض العامة MCU تغطي مجموعة واسعة من تطبيقات التحكم المضمنة مع اتصال عالي السرعة ومعالجة الإشارات الرقمية والتحكم في العاكس (inverter).

تستخدم عائلة متحكم RX بنية محسنة 32 بت من Harvard CISC لتحقيق أداء عالي جدًا.

وظائف الاطراف :




وهو متحكم له 20 دبوس , دبوس 9 هو VSS ، دبوس الأرض و Vdd ، دبوس إمدادات الطاقة. لديها ثلاثة أنواع مختلفة من انظمة اتشغيل او التعليمات ، وهي التعليمة العادية ، التعليمة السريعة ، التعليمة ذات السرعة العالية.

ميزات وفوائد RX Microcontrollers :

- يتم تحقيق انخفاض استهلاك الطاقة باستخدام تكنولوجيا متعددة النواة
- دعم لتشغيل 5 Vللتصاميم الصناعية والأجهزة
- قابلية التوسع من 48 إلى 145 دبوسًا ومن ذاكرة فلاش بسعة 32 كيلو بايت إلى 1 ميجابايت ، مع تضمين ذاكرة فلاش بسعة 8 كيلوبايت
- ميزة السلامة المتكاملة
- مجموعة وظائف غنية متكاملة من 7 UART ، I2C، 8 SPI ، مقارنات ، 12 ADCبت ، 10 DACبت ، 24 ADCبت (RX21A) ، مما سيقلل من تكلفة النظام من خلال دمج معظم الوظائف.

تطبيقات متحكم رينيساس :

| | |
|---|---|
| الاختبار والقياس | الأتمتة الصناعية |
| التطبيقات الطبية | تطبيقات الاتصالات |
| | تطبيقات التحكم في المحركات |

- AVR Microcontrollers

وحدات التحكم الدقيقة AVR هي عبارة عن معمارية هارفارد RISC معدلة مع ذواكر منفصلة للبيانات والبرامج , وسرعة AVR عالية عند مقارنتها بـ 8051 و .PIC يعد AVR مقصورًا على معالج RISC الخاص بـ Alf-Egil Bogen و Vegard .Wollan

الفرق بين 8051 و AVR :

- 8051 هي وحدات تحكم 8 بت تعتمد على بنية CISC لكن AVR هي وحدات تحكم 8 بت تعتمد على بنية RISC
- 8051 يستهلك طاقة أكثر من متحكمAVR
- في 8051 ، يمكننا كتابة البرنامج بسهولة اكثر من متحكم AVR
- سرعة AVR أسرع من متحكم 8051

تصنيف المتحكم AVR :

تصنف AVR Microcontrollers إلى ثلاثة أنواع :

TinyAVR ذاكرة أقل ، حجم صغير ، مناسب فقط للتطبيقات الأكثر بساطة

MegaAVR هذه هي الأكثر شعبية التي لديها كمية جيدة من الذاكرة (تصل إلى 256 كيلو بايت) ، وعدد أكبر من الأجهزة الطرفية يحمل في ثناياه عوامل ومناسبة للتطبيقات المتوسطة إلى المعقدة

XmegaAVR يستخدم تجاريا للتطبيقات المعقدة ، والتي تتطلب ذاكرة برنامج كبيرة وسرعة عالية

ميزات متحكم AVR :

- 16 كيلو بايت فلاش قابلة للبرمجة في النظام
- 512 بايت EEPROM قابلة للبرمجة في النظام
- المؤقت 16 بت مع ميزات إضافية
- مذبذبات داخلية متعددة (Multiple internal oscillators)
- ذاكرة داخلية للتعليمات الذاتية قابلة للبرمجة تصل إلى 256 كيلو بايت مرتبطة في ذاكرة الفلاش
- قابلة للبرمجة داخل النظام باستخدام ISP أو JTAG أو طرق الجهد العالي
- رمز اختياري مع وحدات قفل مستقلة للحماية
- الأجهزة الطرفية التسلسلية المتزامنة / غير المتزامنة (UART / USART)
- ناقل واجهة تسلسلية (SPI)
- واجهة تسلسلية (USI) لنقل البيانات
- مؤقت (WDT)
- انظمة أوضاع النوم الموفرة للطاقة المتعددة
- محولات A / D 10 بت ، مع multiplex حتى 16 قناة
- يمكنه دعم وحدة تحكم ال USB
- الأجهزة ذات الجهد المنخفض تعمل حتى 1.8 فولت

هناك العديد من ميكروكنترولر من الأسرة AVR ، مثل ATmega8 ، ATmega16 وهلم جرا. في هذا الدرس سوف نناقش حول متحكم ATmega328

ATmega328 و ATmega8 هما ICs متوافقان ولكنهما مختلفان وظيفيا. يحتوي ATmega328 على ذاكرة فلاش تبلغ 32 كيلو بايت ، حيث يحتوي ATmega8 على 8 كيلو بايت.

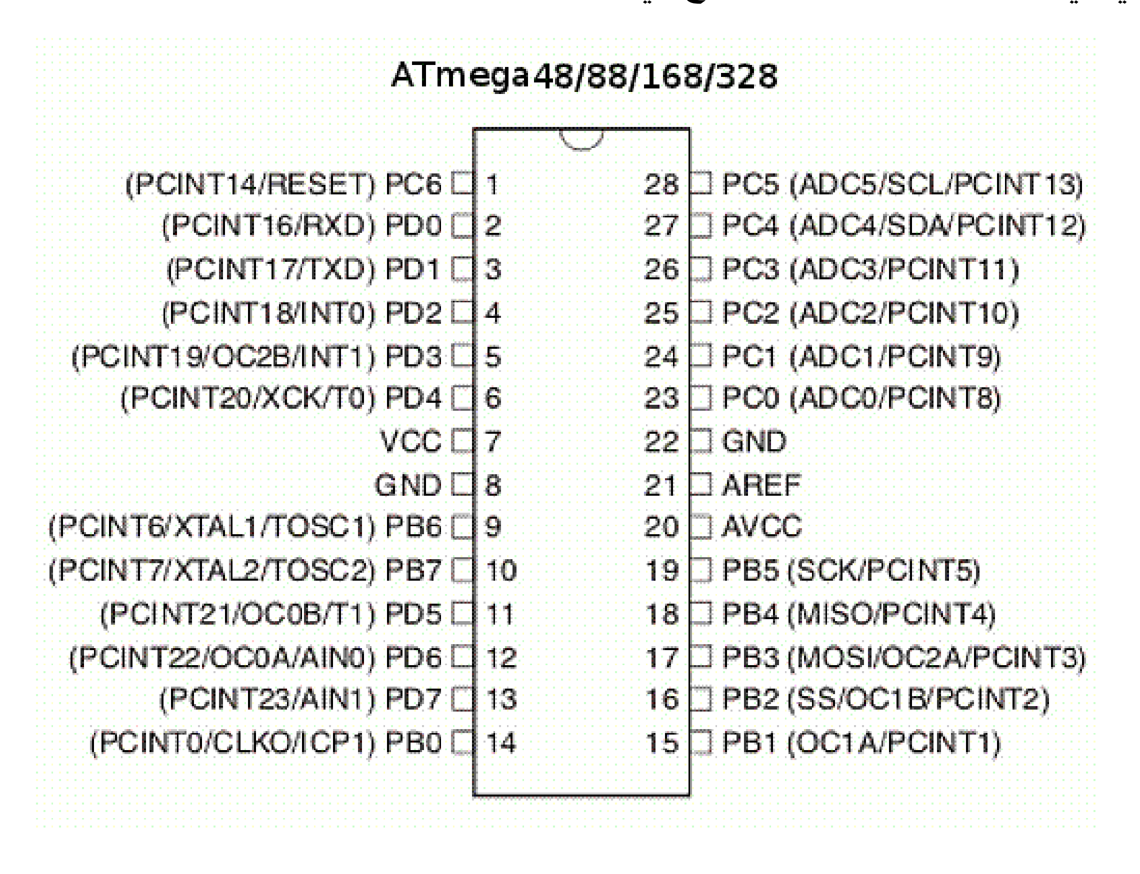
الاختلافات الأخرى هي SRAM إضافية و EEPROM ، إضافة التعليمات تغيير دبوس وأجهزة توقيت وبعض ميزات ATmega328 هي:

## ميزاتATmega328 :

- اثنين من توقيت 8 بت
- محولA / D
- ست قنواتPWM
- في بنيتUSART
- مذبذب خارجي: ما يصل إلى 20MHz
- متحكم AVR ذو 28 سنًا
- ذاكرة فلاش برنامج 32 كيلو بايت
- ذاكرة البيانات EEPROM من 1 كيلو بايت
- ذاكرة البيانات SRAM من 2 كيلو بايت
- دبابيس I / O هي 23

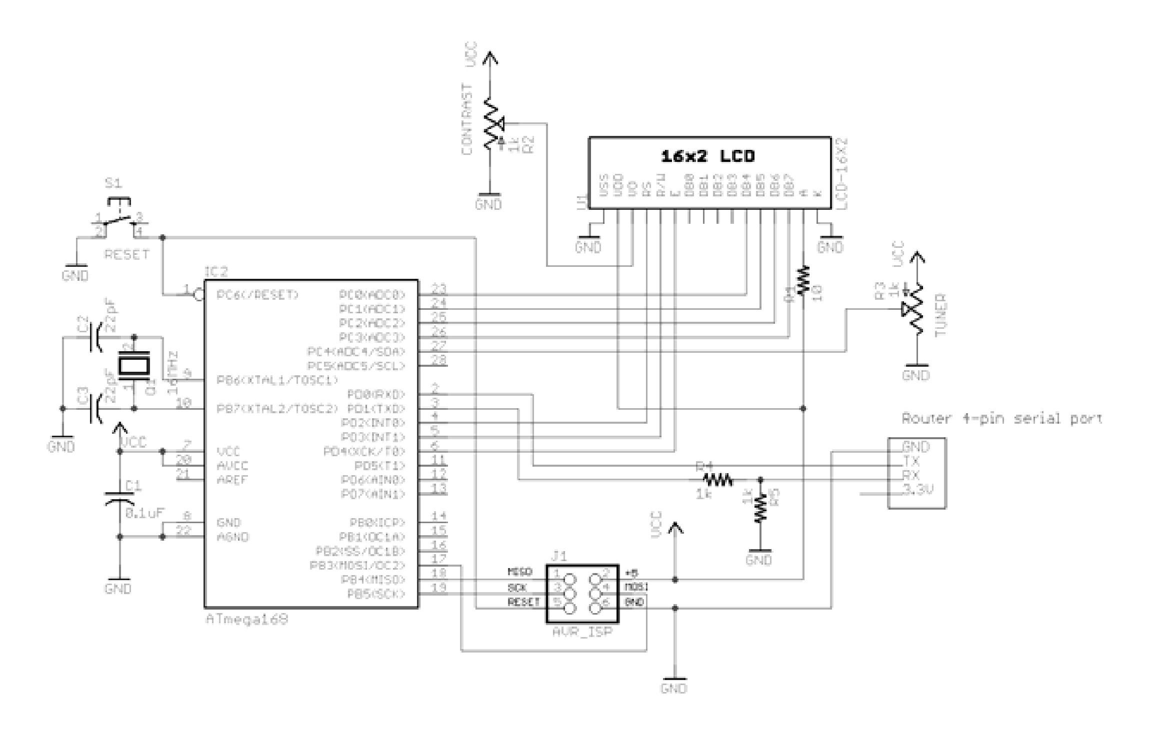
## وظائف اطراف ATmega328:

يأتي في 28 DIP ، كما هو موضح في الشكل أدناه:



| الاطراف | الوظيفة |
|---|---|
| Vcc | Digital supply voltage |
| GND | Ground |
| Port B | Port B is an 8-bit bi-directional I/O port. The Port B pins are tri-stated when a reset condition becomes active or one, even if the clock is not running |
| Port C | Port C is a 7-bit bi-directional I/O port with internal pull-up resistors |
| PC6 | RESET |
| Port D | It is an 8-bit bi-directional I/O port with internal pull-up resistors. The output buffers of the Port D consist symmetrical drive characteristics |
| AVcc | AVcc is the supply voltage pin for the ADC |
| AREF | AREF is the analog reference pin for the ADC |

مثال لدائرة نموذجية من متحكم AVR - ستم توضيحه لاحقًا



تطبيقات متحكم AVR :

هناك العديد من تطبيقات متحكم AVR. يتم استخدامها في أتمتة المنزل ، وشاشة تعمل باللمس ، والسيارات ، والأجهزة الطبية والدفاع.

- PIC Microcontroller

PICهو وحدة تحكم واجهة الطرفية، التي وضعتها الالكترونيات الدقيقة في سنة 1993. يتم التحكم فيه بواسطة البرنامج ويمكن برمجتها لإكمال العديد من المهام والتحكم في خط التوليد وغير ذلك الكثير. تجد ميكروكنترولر PIC طريقها إلى تطبيقات جديدة مثل الهواتف الذكية ، ملحقات الصوت ، ملحقات ألعاب الفيديو والأجهزة الطبية المتقدمة.




مميزات ال PIC :

الميزات الأساسية:

وحدة المعالجة المركزية RISC عالية الأداء

ما يصل إلى 8 كيلو × 14 كلمة من ذاكرة برنامجFLASH

35من  تعليمات (ترميز بطول ثابت - 14 بت)

368×8 ثابت ذاكرة الوصول العشوائي على أساس البيانات

حتى 256 × 8 بايت من ذاكرة بياناتEEPROM

قدرة التعليمات (حتى 14 مصدرًا)

ثلاثة أوضاع عنونة (مباشرة ، غير مباشرة ، نسبية)

إعادة تشغيل الطاقة(POR)

ذاكرة Harvard

توفير الطاقة في وضع النوم

مجموعة واسعة من الجهد التشغيلي : V 2.0إلى V5.5

يمكن ان تعمل على مصدر: 25 ma

يمكن تشغيلها باستخدام البطاريات

الميزات الاضافية :

3 مؤقتات / عدادات ( مسبقة البرمجة ) : Timer0 ، Timer2 8 بت - و Timer1 16 بت يمكن زيادتها أثناء النوم عبر الكريستال الخارجي / ساعة

وحدة مقارنة ، وحدات PWM

- وظيفة التقاط المدخلات تسجيل عدد Timer1 عند انتقال الدبوس

- ناتج وظيفة PWM هو موجة مربعة مع فترة قابلة للبرمجة ودورة العمل.

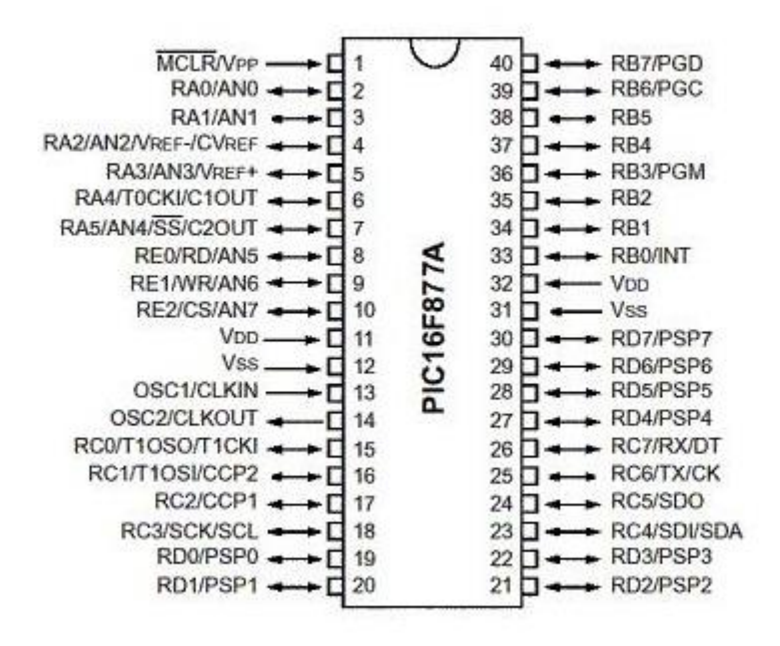
محول 10 بت 8 قنوات تناظرية إلى رقمية

USART مع الكشف عن عنوان 9 بت

منفذ تسلسلي متزامن مع وضع رئيسي وI2C Master/Slave

وحدة المقارنة التناظرية (يمكن مضاعفة المدخلات القابلة للبرمجة من مدخلات الجهاز ومخرجات المقارنة من الخارج)

مخطط وظائف اطراف PIC16F877A :



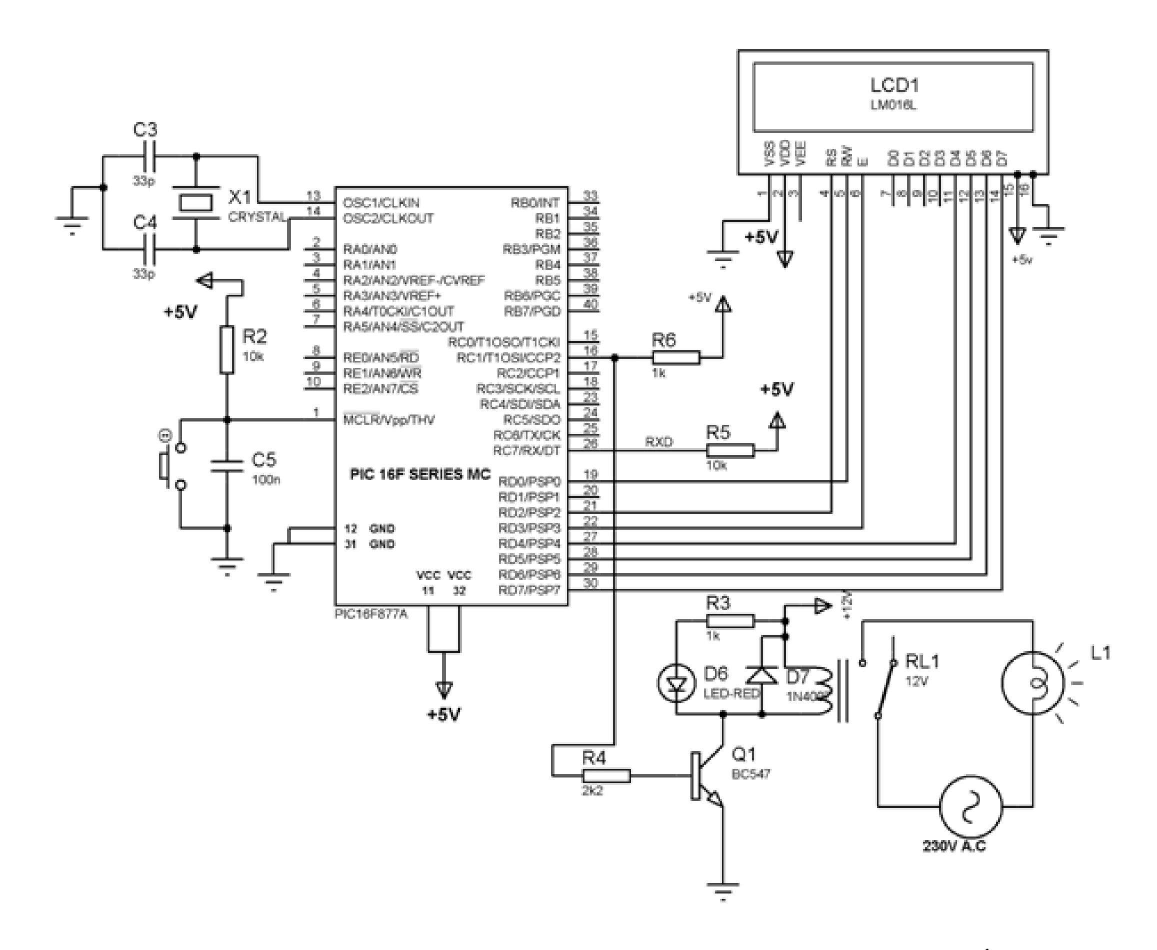

مزايا الموافقة المسبقة عن علم:

تصميمRISC

الكود الخاص به فعال للغاية ، مما يسمح لـ PIC بالعمل بذاكرة برنامجية أقل عادة من منافسيها الأكبر

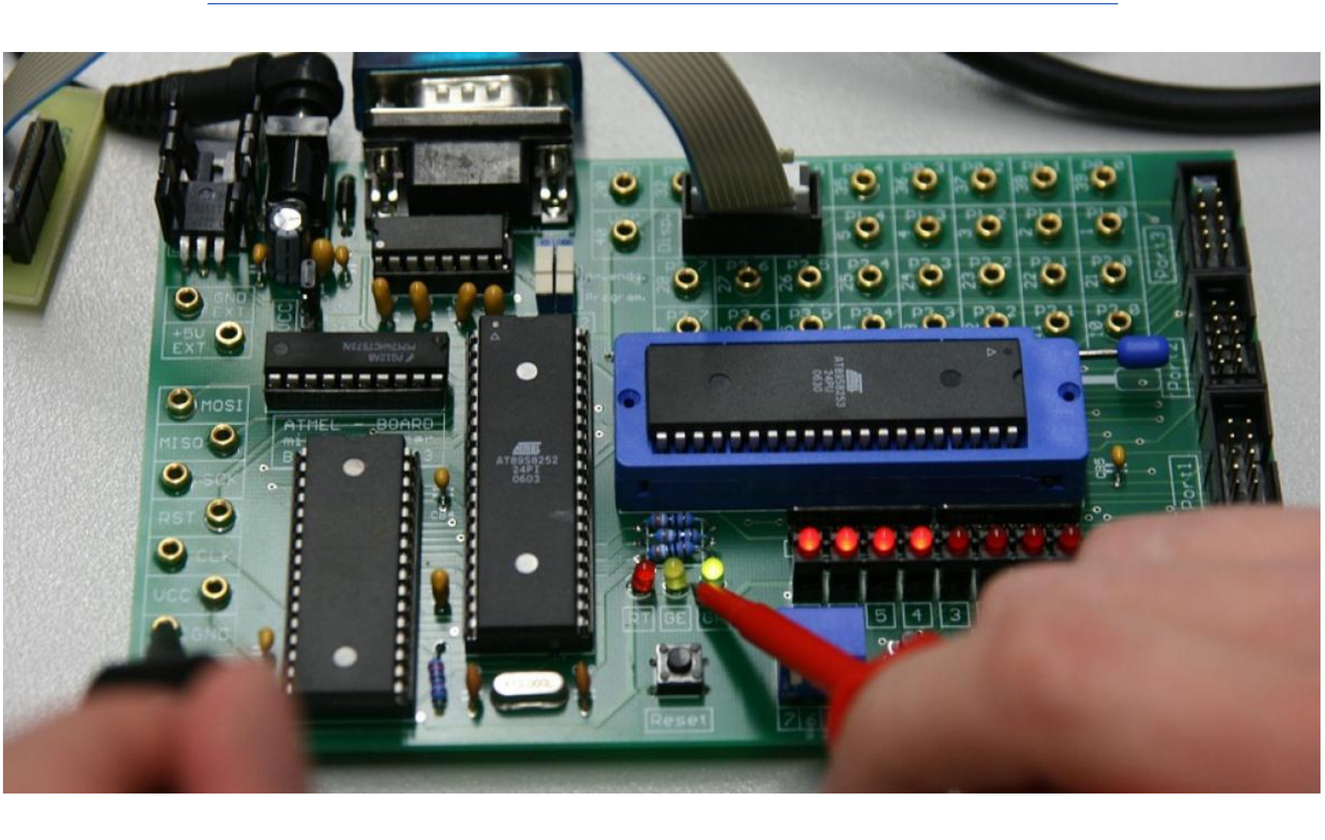
أنها منخفضة التكلفة ، وارتفاع سرعة الساعة

دارة تطبيق نموذجية لـ PIC16F877A (مثال توضيحي لاستخدامات PIC) :



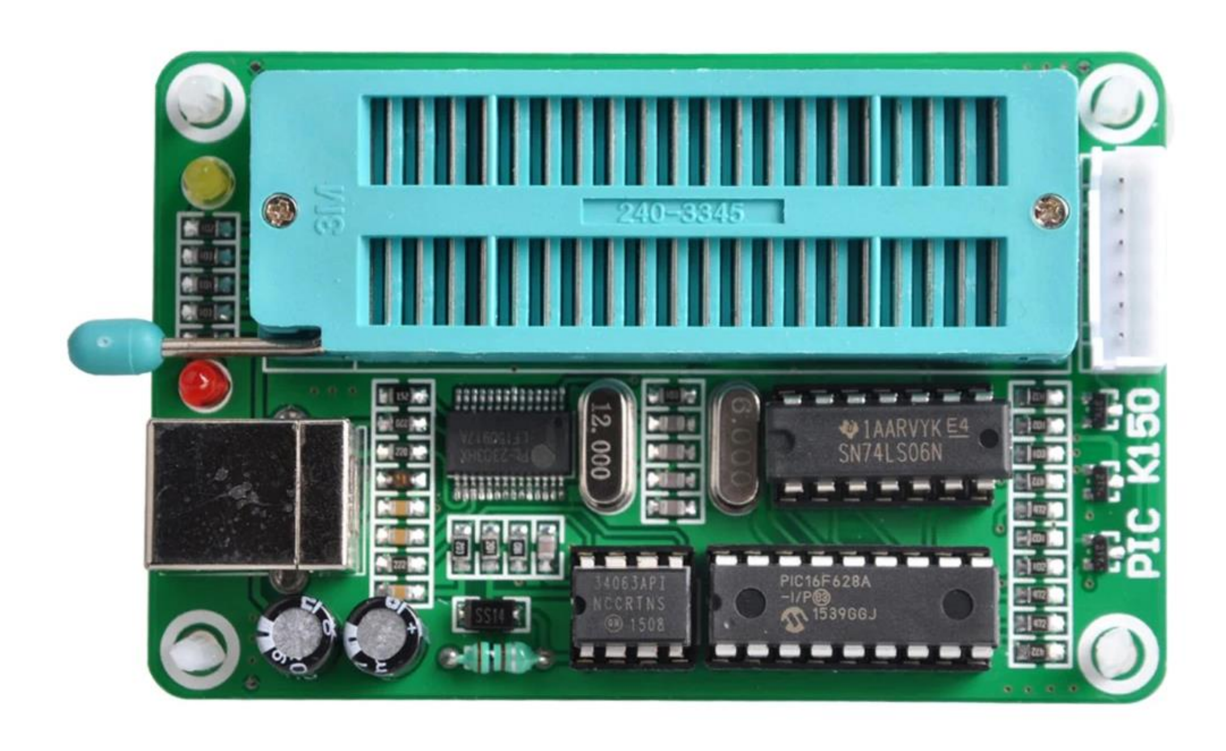

تتكون الدائرة أعلاه من مصباح يتم التحكم بتبديله باستخدام متحكم PIC يتم التحكم في Microcontroller مع الكريستال الخارجي الذي يوفر مدخلات على مدار الساعة. يتم ربط PIC أيضًا مع زر ضغط وعند الضغط على زر الضغط ، وبناءً على ذلك ، يرسل Microcontroller إشارة عالية إلى قاعدة الترانزستور ، وذلك لتشغيل الترانزستور ومن ثم توفير اتصال مناسب للمرحل لتشغيله و السماح بمرور التيار المتردد إلى المصباح وبالتالي يضيء المصباح.

ويتم عرض حالة العملية على شاشة LCD التي يتم ربطها بوحدة التحكم.PIC




حسنا بعد ان درسنا المكونات الداخلية للميكرو كونترولر وتعرفنا على انواعه المختلفة واطرافه وتطبيقاته بقي لنا ان نبدأ العمل بها اي نبدا ببرمجة المتحكم , لكن هل تسائلت كيف يتم ربط الميكرو كونترولر بجهاز الحاسوب حتى يبدأ المتحكم في استقبال الاوامر (البرنامج) الذي به سيعمل ويؤدي الوظيفة المطلوبة, في هذا الدرس سوف ندرس الاداة او الجهاز المسؤول عن ربط الميكرو كونترولر بالحاسوب يوجد اشكال وانواع واصدارات مختلفة لها فمنها محدود بعدد الاطراف ومنها الكبير جدا ومنها السريعة والبطيئة ... الخ من خصائص سيتم دراستها.

## ▪ K150 PIC Programmer




تحتوي هذه المبرمجة على رقم البروتوكول للمساعدة في مطابقة الأجهزة والبرامج الثابتة تدعم معظم معالجات PIC ولا تحتاج إلى تغذية خارجيةوهي سرعة في عملية البرمجة وتدعم تدعم ZIF ذات 40 رجل -ووصلةICSP متوافقة مع نظام التشغيل XP/7/8/8.1

تتميز بسهولة في الاستخدام وبرنامج التشغيل بسيط و تعمل عبر منفذ USB

تدعم هذه المبرمجة المتحكمات التالية :

| | |
|---|---|
| 12C508 16C65A 16C77 16F76 16F877 | 12CE673 16C620A 16C765 16F83 18F442 |
| 12C508A 16C65B 16C710 16F77 16F877A | 12CE674 16C621 16C773 16F84 18F448 |
| 12C509 16C66 16C711 16F737 18F242 | 12F62916C621A 16C774 16F84A 18F452 |
| 12C509A 16C66A 16C712 16F747 18F248 | 12F675 16C622 16C83 16F87 18F458 |
| 12C671 16C67 16C716 16F767 18F252 | 16C505 16C622A 16C84 16F88 18F1220 |
| 12C672 16C620 16C745 16F777 18F258 | 16C554 16C71 16F627 16F818 18F1320 |

16C558 16C71A 16F627A 16F819
18F2220

16C61 16C72 16F628 16F870 18F2320

16C62 16C72A 16F628A 16F871 18F4220

16C62A 16C73 16F630 16F872 18F4320

16C62B 16C73A 16F648A 16F873 16C63

16C73B 16F676 16F873A

Added from diypack23:

16C63A 16C74 16F684 16F874 16F5x

16C64 16C74A 16F688 16F874A 10Fxxx

18F6525 6621 8525 8621

(all beta) 16C64A 16C74B 16F73 16F876

16C65 16C76 16F74 16F876A

Added diypack25 12F683

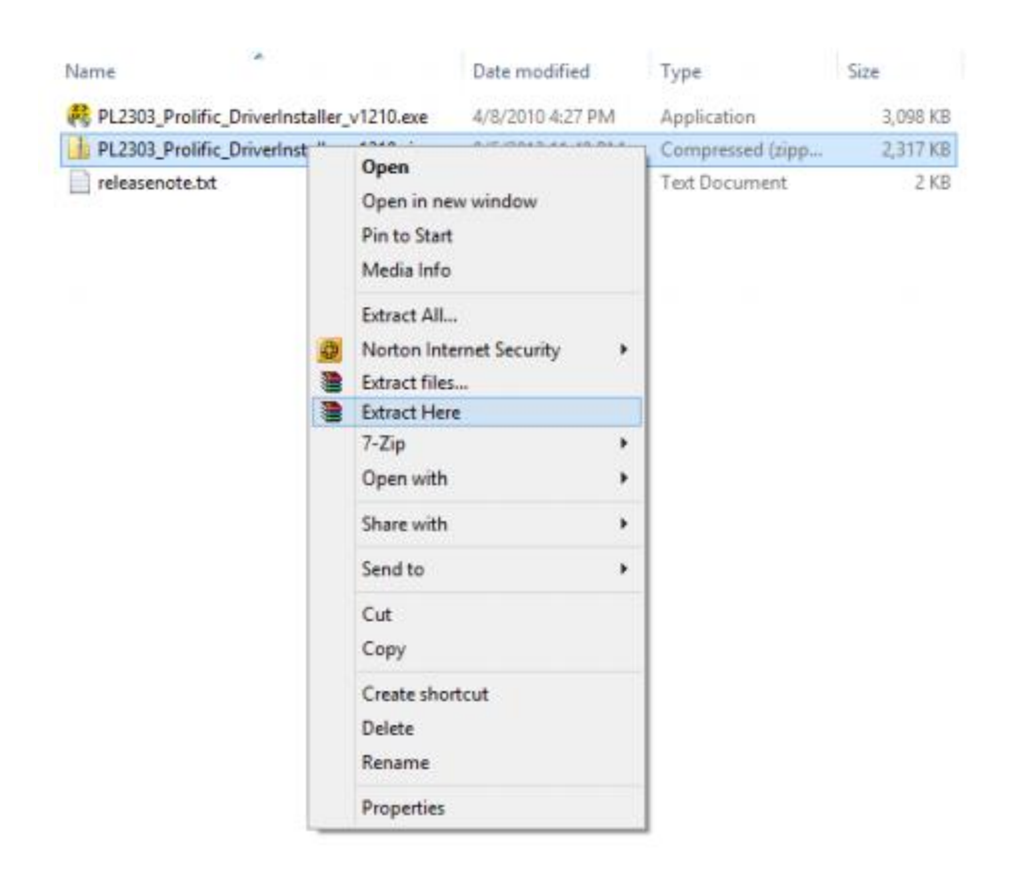
يتم ربط القطعة مع الحاسوب بواسطة كيبل USB مثل الذي موضح بالصوة :

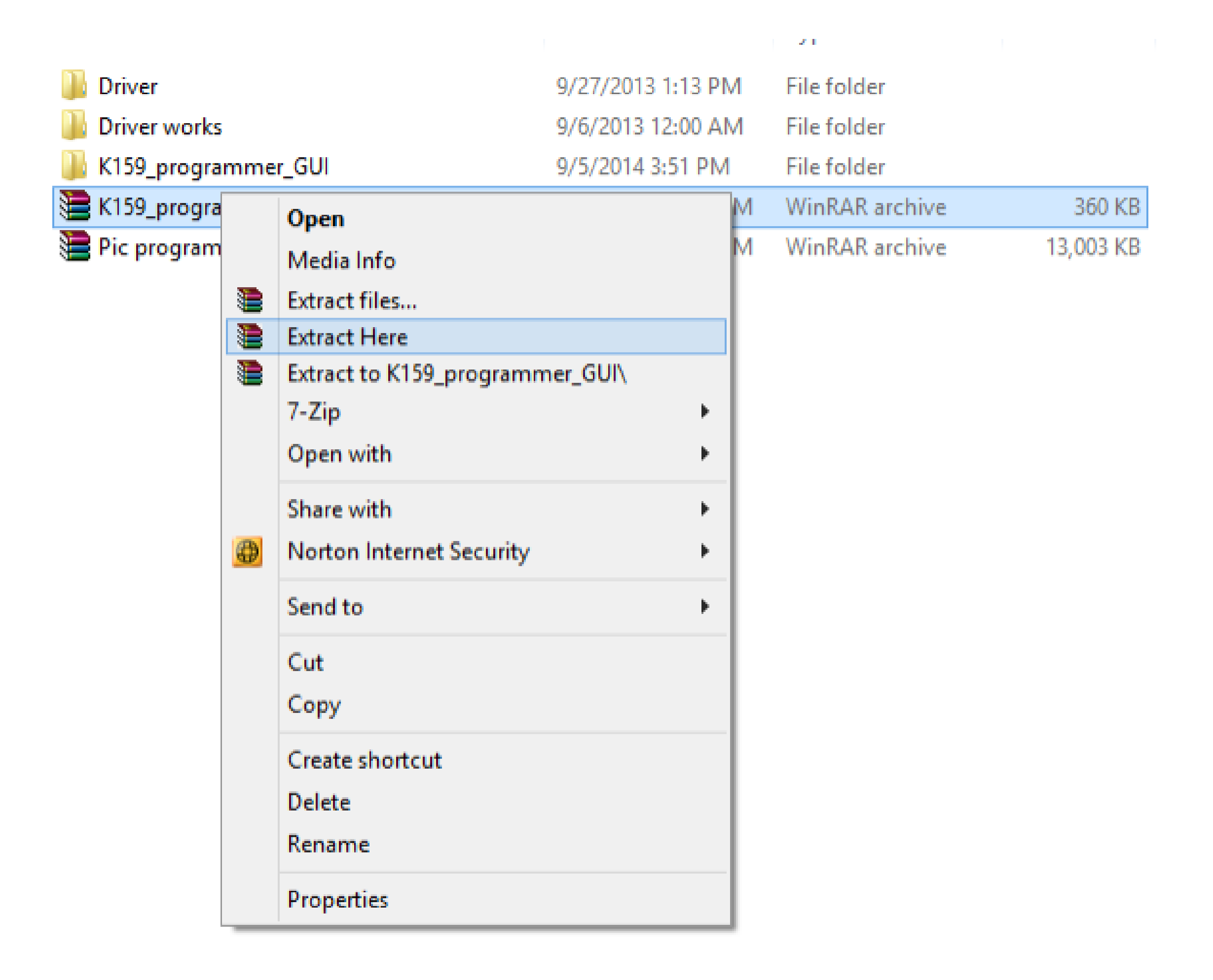


الان تقوم بتحميل البرنامج المسؤول على تعريف القطعة وعملية البرمجة من خلال الرابط التالي , او من محتويات القرص

http://kitsrus.com/pdf/pic.html

# تنصيب البرنامج

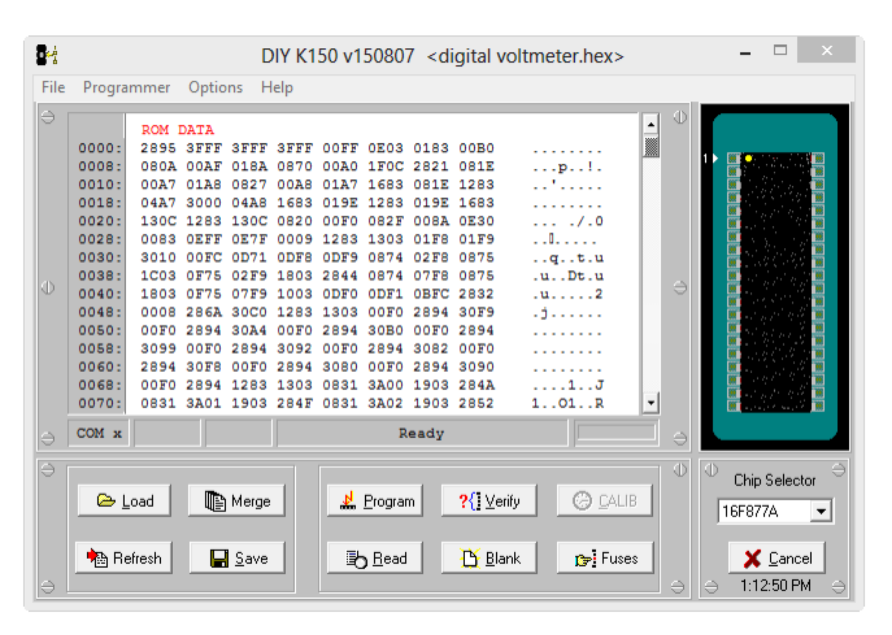




Name
Date modified
Type
Size
PL2303_Prolific_DriverInstaller_v1210.exe
4/8/2010 4:27 PM
Application
3,098 KB
PL2303_Prolific_DriverInstaller_v1210.zip
9/5/2013 11:48 PM
Compressed (zipp...
2,317 KB
releasenote.txt
4/13/2010 11:39 AM
Text Document
2 KB
Driver
9/27/2013 1:13 PM
File folder
Driver works
9/6/2013 12:00 AM
File folder
K159_programmer_GUI
9/5/2014 3:51 PM
File folder
WinRAR archive
360 KB
WinRAR archive
13,003 KB
Open
Media Info
Extract files...
Extract Here
Extract to K159_programmer_GUI\
7-Zip
Open with
Share with
Norton Internet Security
Send to
Cut
Copy
Create shortcut
Delete
Rename
Properties

| | | | |
|---|---|---|---|
| Driver | 9/27/2013 1:13 PM | File folder | |
| Driver works | 9/6/2013 12:00 AM | File folder | |
| K159_programmer_GUI | 9/5/2014 3:51 PM | File folder | |
| K159_programmer_GUI.rar | 11/11/2013 9:18 PM | WinRAR archive | 360 KB |
| Pic programmer.rar | 11/10/2013 5:03 PM | WinRAR archive | 13,003 KB |

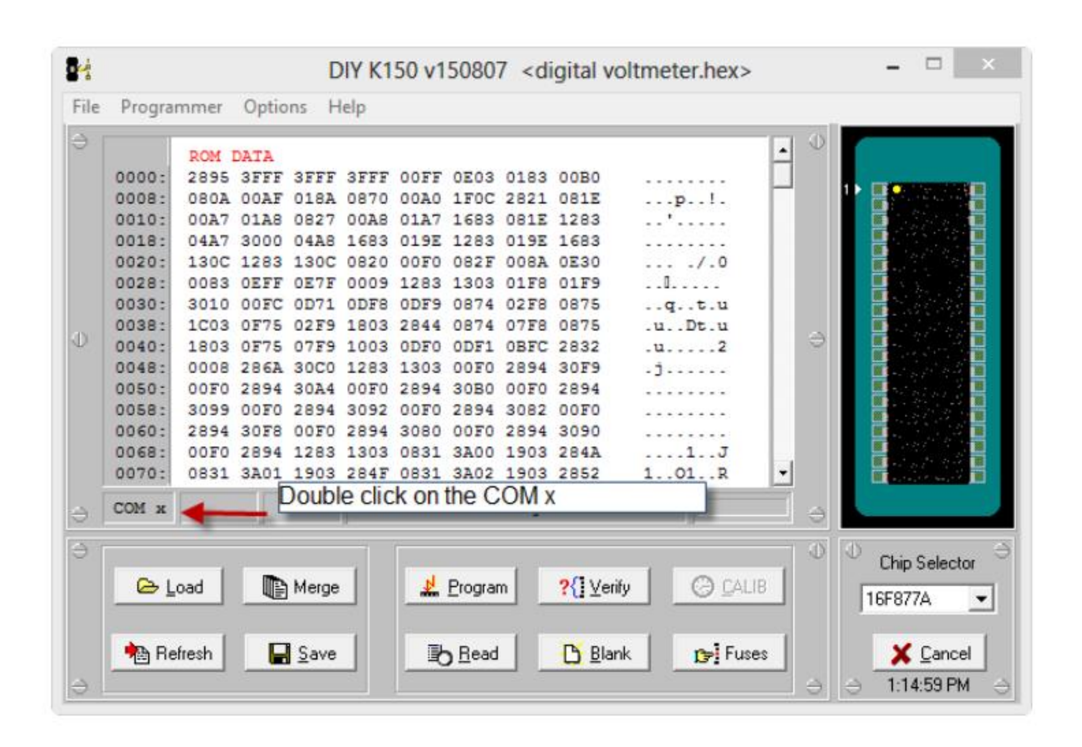
| | | | |
|---|---|---|---|
| chipdata.cid | 8/23/2007 5:06 PM | CID File | 200 KB |
| DIYpack25ep.zip | 9/2/2013 3:28 PM | Compressed (zipp... | 405 KB |
| epk128.hex | 9/5/2007 12:52 PM | HEX File | 12 KB |
| epk149a.hex | 9/5/2007 12:52 PM | HEX File | 12 KB |
| epk149bf.hex | 9/5/2007 12:52 PM | HEX File | 12 KB |
| epk150.hex | 9/5/2007 12:52 PM | HEX File | 12 KB |
| epk182.hex | 9/5/2007 12:52 PM | HEX File | 12 KB |
| fixhex2.exe | 9/12/2007 2:56 PM | Application | 22 KB |
| microbrn.exe | 8/23/2007 5:06 PM | Application | 660 KB |
| microbrn.hlp | 8/23/2007 6:16 PM | Help file | 47 KB |
| pro.dat | 9/22/2014 12:39 AM | DAT File | 1 KB |
| readme_ep.pdf | 9/11/2007 10:30 PM | Adobe Acrobat D... | 44 KB |



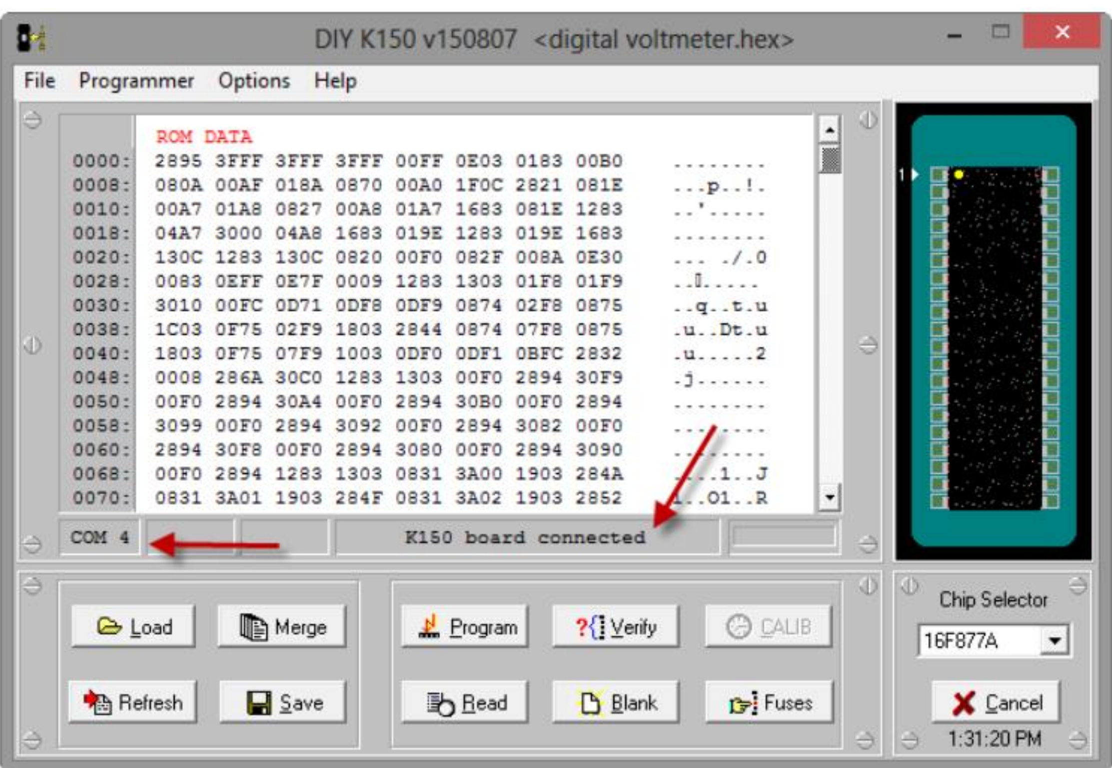
نختار اللوحة

نعرف المخرج

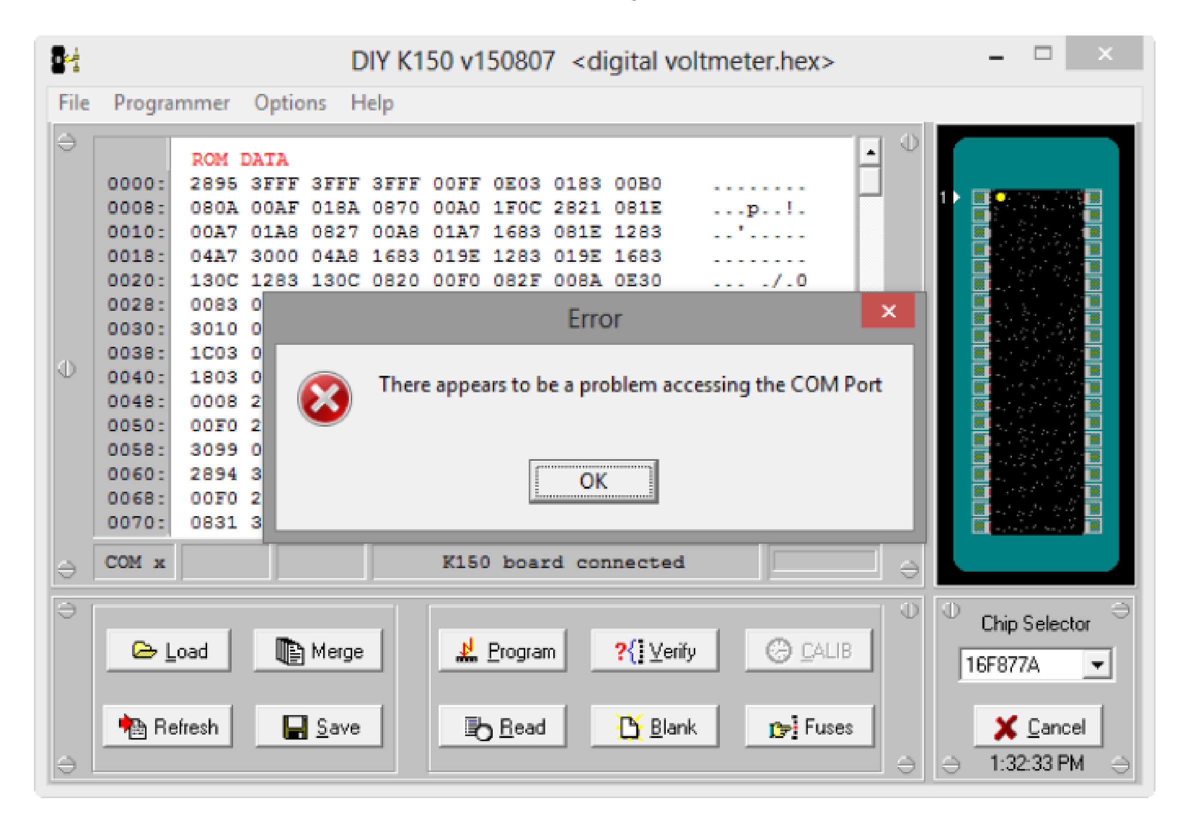



DIY K150 v150807 <digital voltmeter.hex>
File
Programmer
Options
Help
ROM DATA
COM x
Double click on the COM x
Load
Merge
Program
Verify
CALIB
Refresh
Save
Read
Blank
Fuses
Chip Selector
16F877A
Cancel
1:14:59 PM

DIY K150 v150807 <digital voltmeter.hex>
File Programmer Options Help
ROM DATA
0000: 2895 3FFF 3FFF 3FFF 00FF 0E03 0183 00B0
0008: 080A 00AF 018A 0870 00A0 1F0C 2821 081E ...p..!.
0010: 00A7 01A8 0827 00A8 01A7 1683 081E 1283
0018: 04A7 3000 04A8 1683 019E 1283 019E 1683
0020: 130C 1283 130C
0028: 0083 0EFF 0E7F
0030: 3010 00FC 0D71
0038: 1C03 0F75 02F9
0040: 1803 0F75 07F9
0048: 0008 286A 30C0
0050: 00F0 2894 30A4
0058: 3099 00F0 2894
0060: 2894 30F8 00F0
0068: 00F0 2894 1283
0070: 0831 3A01 1903
Kitsrus.com Serial Port Change
Enter Port Value 4
OK Cancel
COM x Ready
Load Merge Program Verify CALIB
Refresh Save Read Blank Fuses
Chip Selector
16F877A
Cancel
1:30:07 PM



اذا ظهر هذا الخطأ :




بحيث يكون لديك منفذ COM خاطئ أو لم تقم بتثبيت برنامج التشغيل بشكل صحيح

تحميل ملف Hex والبرمجة:

DIY K150 v150807 <digital voltmeter.hex>

File Programmer Options Help

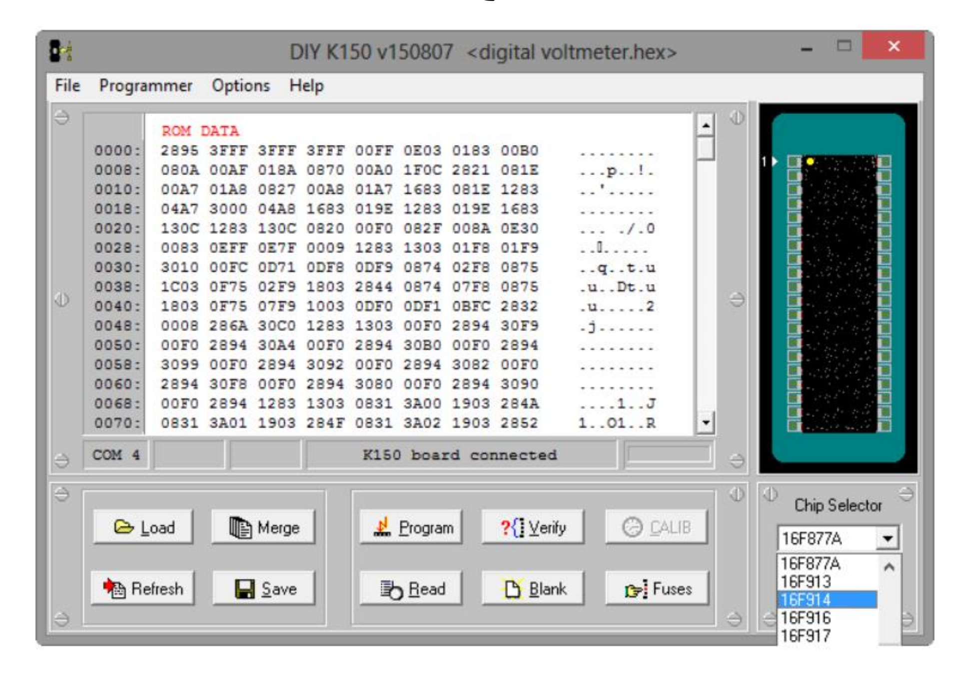
```
ROM DATA
0000: 2895 3FFF 3FFF 3FFF 00FF 0E03 0183 00B0   ........
0008: 080A 00AF 018A 0870 00A0 1F0C 2821 081E   ...p..!.
0010: 00A7 01A8 0827 00A8 01A7 1683 081E 1283   ..'.....
0018: 04A7 3000 04A8 1683 019E 1283 019E 1683   ........
0020: 130C 1283 130C 0820 00F0 082F 008A 0E30   ... ./.0
0028: 0083 0EFF 0E7F 0009 1283 1303 01F8 01F9   .......
0030: 3010 00FC 0D71 0DF8 0DF9 0874 02F8 0875   ..q..t.u
0038: 1C03 0F75 02F9 1803 2844 0874 07F8 0875   .u..Dt.u
0040: 1803 0F75 07F9 1003 0DF0 0DF1 0BFC 2832   .u.....2
0048: 0008 286A 30C0 1283 1303 00F0 2894 30F9   .j......
0050: 00F0 2894 30A4 00F0 2894 30B0 00F0 2894   ........
0058: 3099 00F0 2894 3092 00F0 2894 3082 00F0   ........
0060: 2894 30F8 00F0 2894 3080 00F0 2894 3090   ........
0068: 00F0 2894 1283 1303 0831 3A00 1903 284A   ....1..J
0070: 0831 3A01 1903 284F 0831 3A02 1903 2852   1..O1..R
```

COM 4 K150 board connected

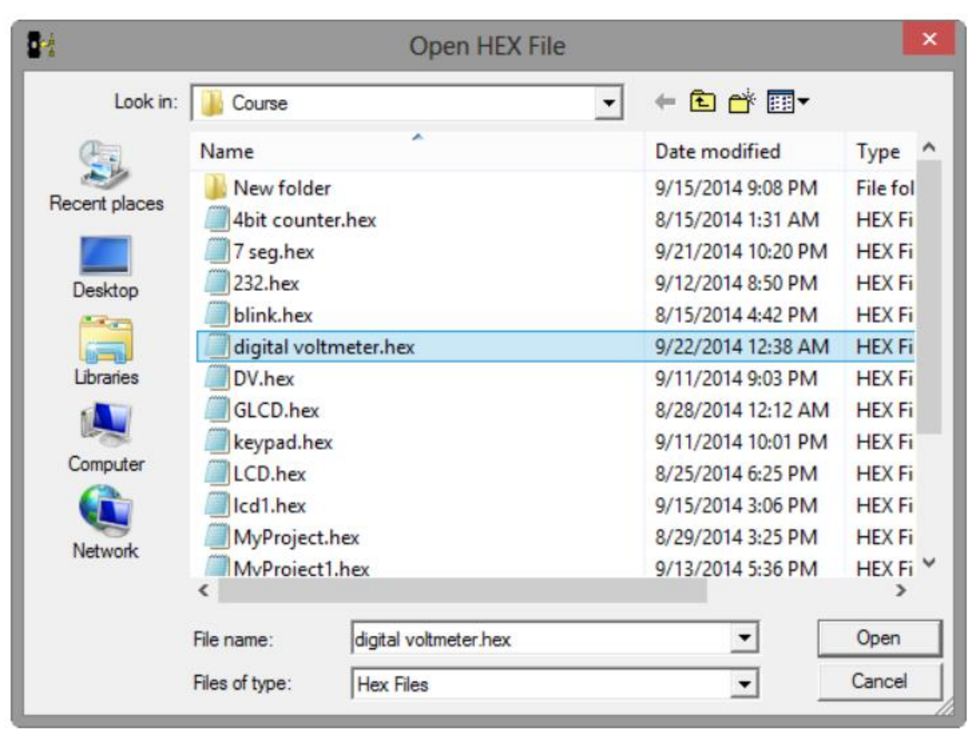
Load Merge Program Verify CALIB Chip Selector 16F877A

Refresh Save Read Blank Fuses Cancel 1:35:11 PM

اختار نوع الشريحة



ثم قم بتحميل ملف HEX الخاص بك (بعد اعداد البرنامج كما سندرسه لاحقا)



DIY K150 v150807 <digital voltmeter.hex>

File Programmer Options Help

```
ROM DATA
0000: 2895 3FFF 3FFF 3FFF 00FF 0E03 0183 00B0
0008: 080A 00AF 018A 0870 00A0 1F0C 2821 081E
0010: 00A7 01A8 0827 00A8 01A7 1683 081E 1283
0018: 04A7 3000 04A8 1683 019E 1283 019E 1683
0020: 130C 1283 130C 0820 00F0 082F 008A 0E30
0028: 0083 0EFF 0E7F 0009 1283 1303 01F8 01F9
0030: 3010 00FC 0D71 0DF8 0DF9 0874 02F8 0875
0038: 1C03 0F75 02F9 1803 2844 0874 07F8 0875
0040: 1803 0F75 07F9 1003 0DF0 0DF1 0BFC 2832
0048: 0008 286A 30C0 1283 1303 00F0 2894 30F9
0050: 00F0 2894 30A4 00F0 2894 30B0 00F0 2894
0058: 3099 00F0 2894 3092 00F0 2894 3082 00F0
0060: 2894 30F8 00F0 2894 3080 00F0 2894 3090
0068: 00F0 2894 1283 1303 0831 3A00 1903 284A
0070: 0831 3A01 1903 284F 0831 3A02 1903 2852
```

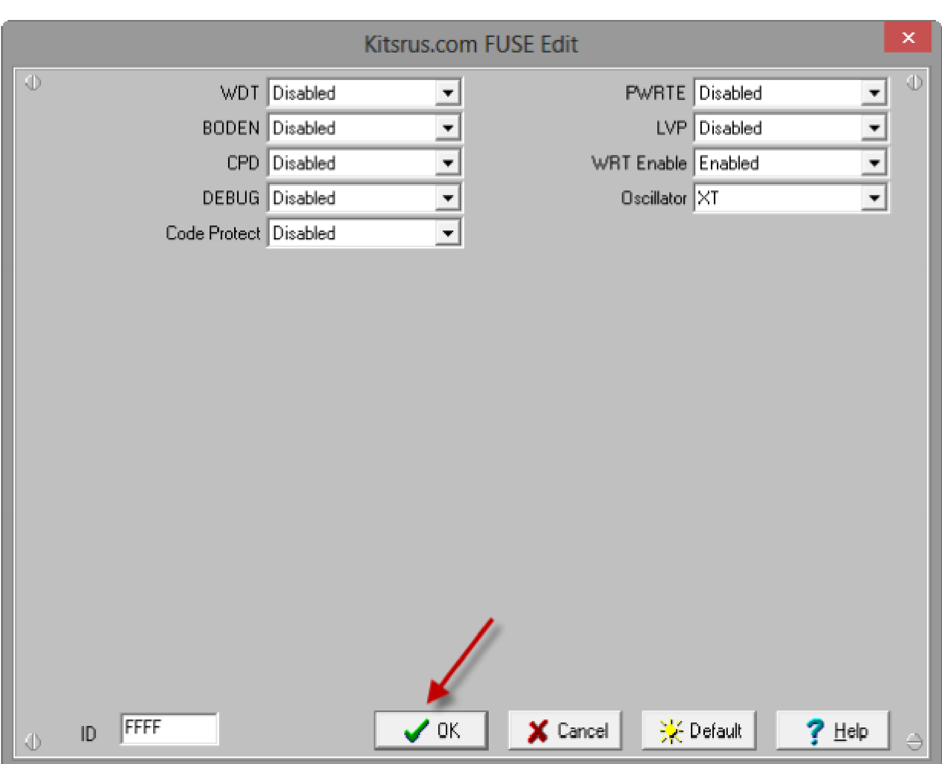
COM 4 Ready

Load Merge Program Verify CALIB

Refresh Save Read Blank Fuses

Chip Selector 16F877A

Cancel 1:38:15 PM



DIY K150 v150807 <digital voltmeter.hex>

File Programmer Options Help

ROM DATA

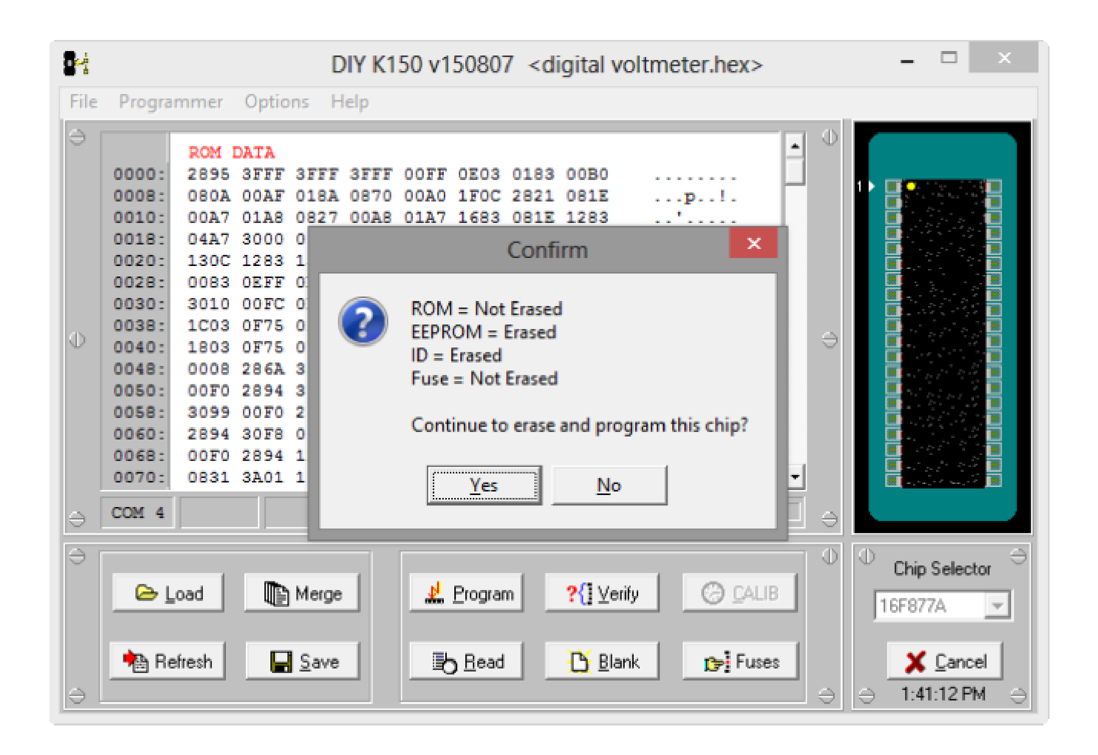
```
0000: 2895 3FFF 3FFF 3FFF 00FF 0E03 0183 00B0   ........
0008: 080A 00AF 018A 0870 00A0 1F0C 2821 081E   ...p..!.
0010: 00A7 01A8 0827 00A8 01A7 1683 081E 1283   ..'.....
0018: 04A7 3000 04A8 1683 019E 1283 019E 1683   ........
0020: 130C 1283 130C 0820 00F0 082F 008A 0E30   ... ./.0
0028: 0083 0EFF 0E7F 0009 1283 1303 01F8 01F9   ........
0030: 3010 00FC 0D71 0DF8 0DF9 0874 02F8 0875   ..q..t.u
0038: 1C03 0F75 02F9 1803 2844 0874 07F8 0875   .u..Dt.u
0040: 1803 0F75 07F9 1003 0DF0 0DF1 0BFC 2832   .u.....2
0048: 0008 286A 30C0 1283 1303 00F0 2894 30F9   .j......
0050: 00F0 2894 30A4 00F0 2894 30B0 00F0 2894   ........
0058: 3099 00F0 2894 3092 00F0 2894 3082 00F0   ........
0060: 2894 30F8 00F0 2894 3080 00F0 2894 3090   ........
0068: 00F0 2894 1283 1303 0831 3A00 1903 284A   .....1..J
0070: 0831 3A01 1903 284F 0831 3A02 1903 2852   1..O1..R
```

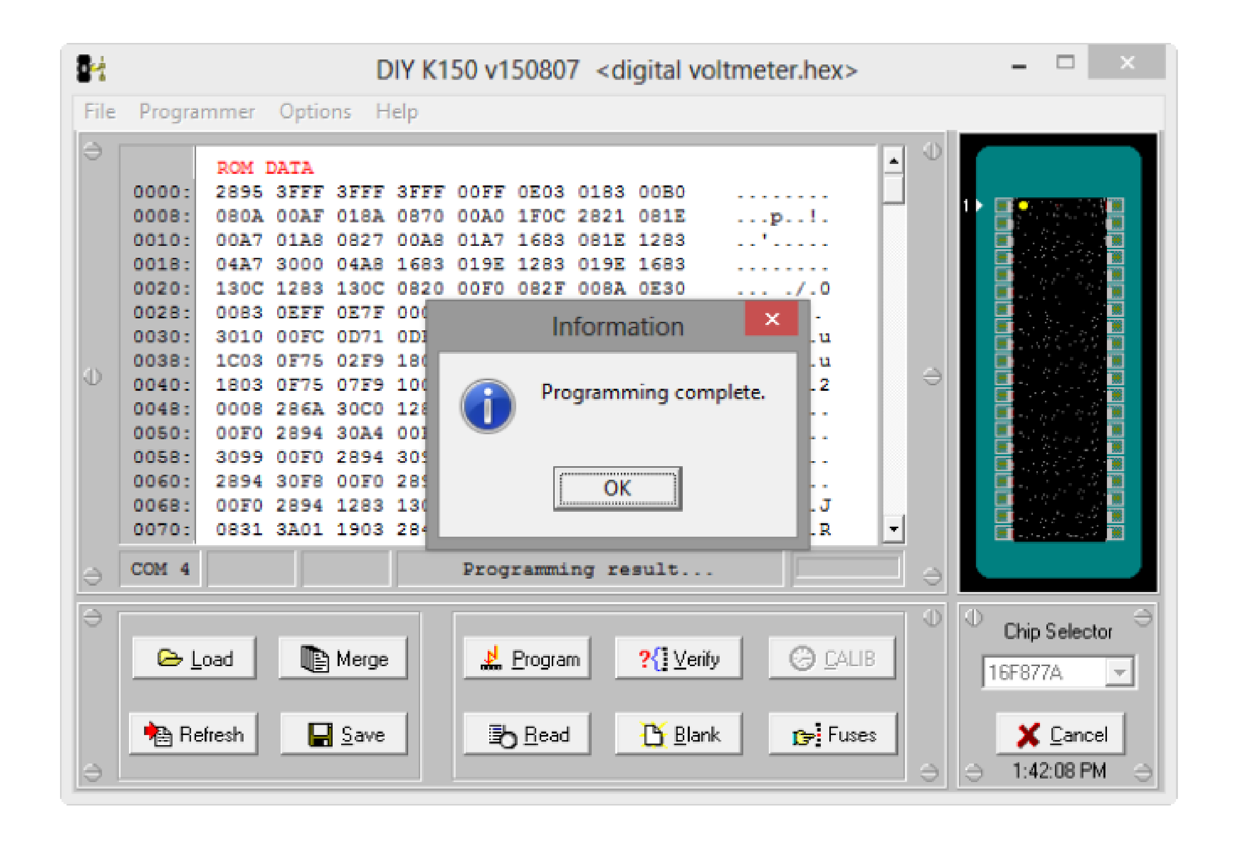
COM 4 Ready

Load Merge Program Verify CALIB

Refresh Save Read Blank Fuses

Chip Selector 16F877A Cancel 1:40:18 PM






وهكذا نكون قمنا بتحميل البرنامج الى الميكرو كونترولر

- pickit3




البرمجة Pickit3 يأتي مع المكونات التالية :

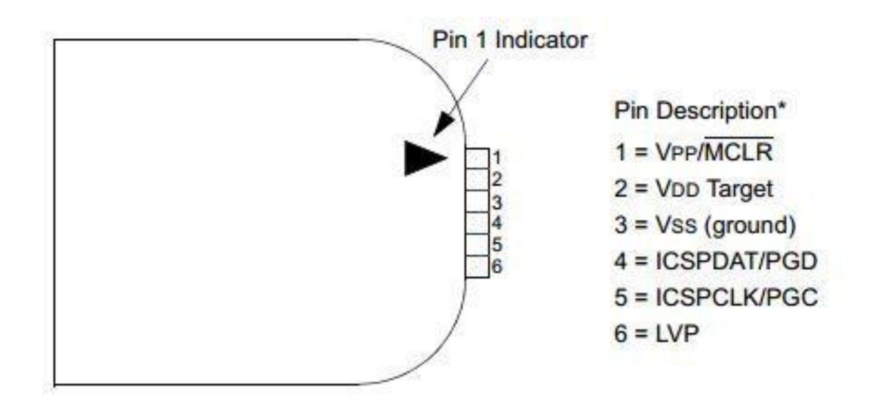
- Lanyard Connection
- USB Port Connection
- Pin 1 Marker
- Programming connector
- Status LEDs
- Push Button




Lanyard Connection: للربط بين Pickit و القاعدة

Pin 1 Marker: أشارت إلى موقع الدبوس 1 للتوصيل الصحيح مع الحد الأدنى من لوحة التطوير التي يتم وضع وحدة التحكم الدقيقة فيها PIC.

Programming Connector: مخرج موصل ستة اطراف ويستخدم لتوصيل المتحكم مع PICKit3. وتتكون من الدبابيس التالية ابتداءا من العلامة 1 :



يتم توفير عدة مصابيح LED على PICKit3 تشير الألوان المختلفة إلى حالة مختلفة لـ PICKit3 كما يلي:

Power(Green) Power is supplied to the PICKit3 through USB port.

Active (Blue) Communication link is active and PICKit3 has connection with PC through USB cable.

Status Busy (Yellow) Some function is in progress and PICKit3 is busy with it like programming

Error (Red) The PICKit3 encounter some error.

الاتصال مع PIC Microcontroller:

لتوصيل PICKit3 مع PIC Microcontrollers تستخدم الطريقة التالية حسب ورقة البيانات المرفقة :

Pin 1 of burner with Pin4 of PIC Microcontroller

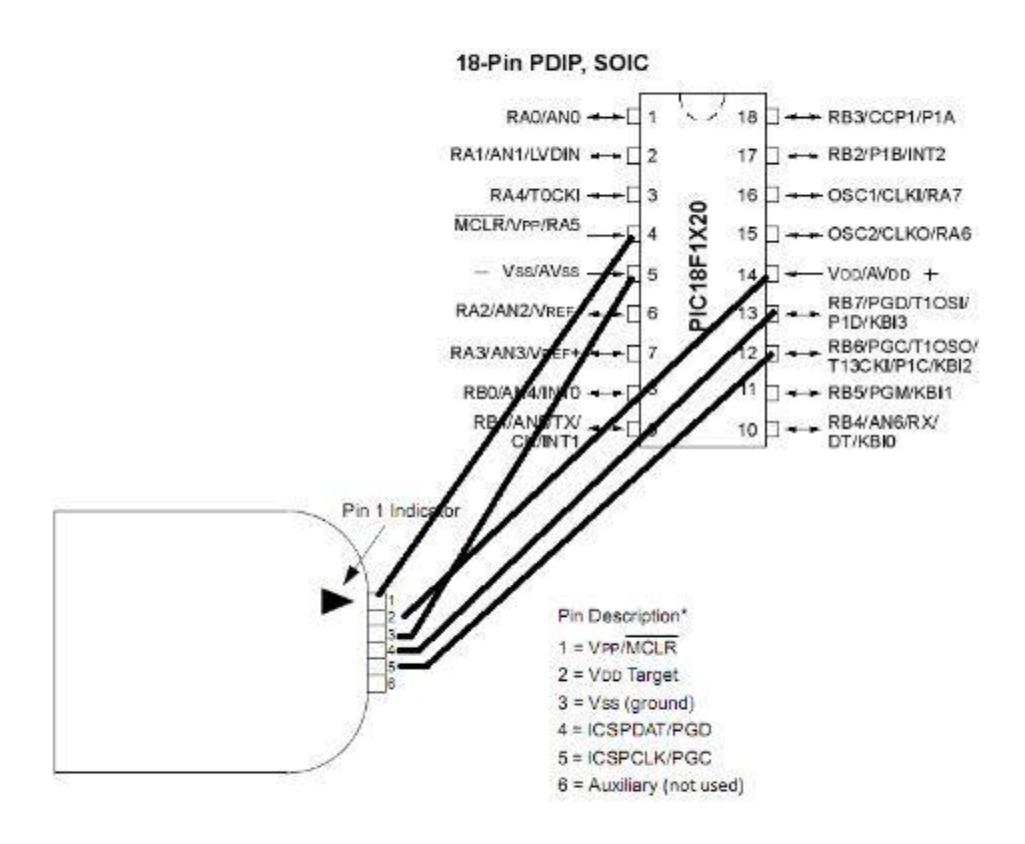
Pin 2 of burner with Pin14 of PIC Microcontroller

Pin 3 of burner with Pin 5 of PIC Microcontroller

Pin 4 of burner with Pin 13 of PIC Microcontroller

Pin 5 of burner with Pin 12 of PIC Microcontroller

Pin 6 is not connected for normal usage.



هذا التركيب مخصص فقط لـ **PIC18F1x20** لكل تركيب اطراف microcontroller سيكون مختلفا ويمكنك استشارة ورقة البيانات الخاصة بوحدة التحكم الدقيقة الخاصة بك لهذه التركيبات !.

الوحدة الطرفية التي بها نربط المتحكم

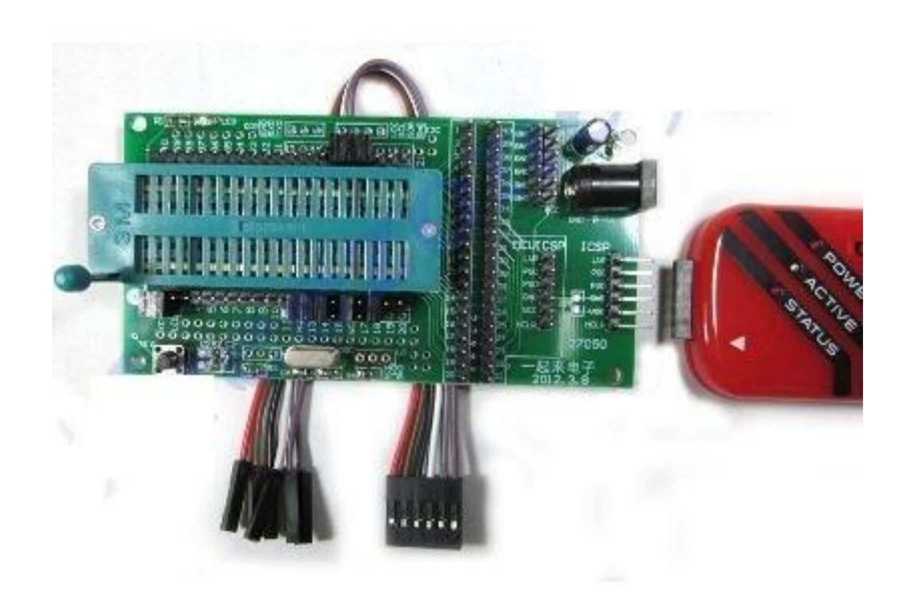


1. Vpp/MCLR (Power)
2. $V_{DD}$ Target (Power on Target)
3. $V_{SS}$ (Ground)
4. ICSPDAT/PGD (Standard COM Data)
5. ICSPCLK/PGC (Standard COM Clock)
6. LVP (Low Voltage Programming)

توصيل لوحة مبرمج مع PICKit3:

1. Pin 1 of PICKit3 with MCLR of Programmer Board
2. Pin 2 of PICKit3 with VCC of Programmer Board
3. Pin 3 of PICKit3 with GND of Programmer Board
4. Pin 4 of PICKit3 with PGD of Programmer Board
5. Pin 5 of PICKit3 with PGC of Programmer Board

Pin 6 is not connected for normal usage.



## توصيل وحدة التحكم الدقيقة بلوحة مبرمج

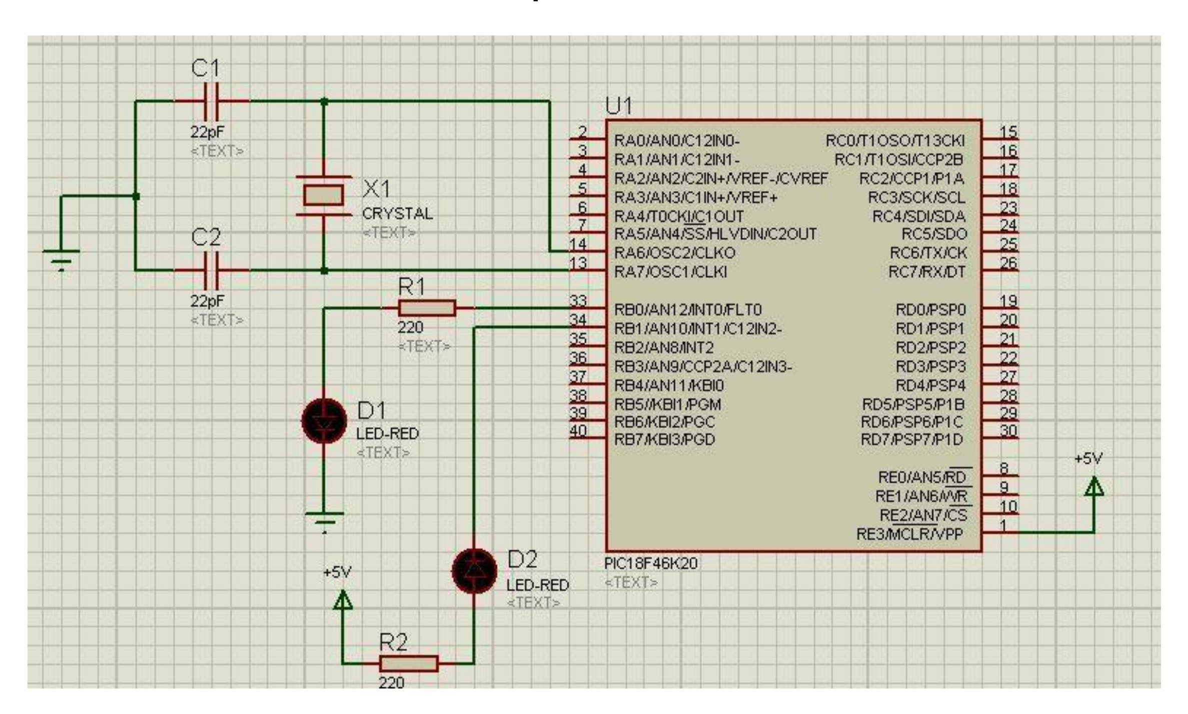
1. MCLR of Programmer Board with Pin 1 of PIC18F46K22
2. VCC of Programmer Board with Pin 11 of PIC18F46K22
3. GND of Programmer Board with Pin 12 of PIC18F46K22
4. PGD of Programmer Board with Pin 40 of PIC18F46K22
5. PGC of Programmer Board with Pin 39 of PIC18F46K22

Pin 6 is not connected for normal usage

هذه الاتصالات مخصصة فقط لـ PIC18F46K22. بالنسبة للميكروكونترولر الأخرى ، راجع ورقة البيانات الخاصة بوحدة التحكم الدقيقة.

الان سوف اقدم مثال هو فقط من اجل توضيح عملية البرمجة وكيفية التعامل مع البرامج المختلفة للبرمجة , ليس من الضروري ان تفهم الكود او القطع في المخطط , سوف اقوم بشرحها مفصلا في الدروس القادمة

المخطط :



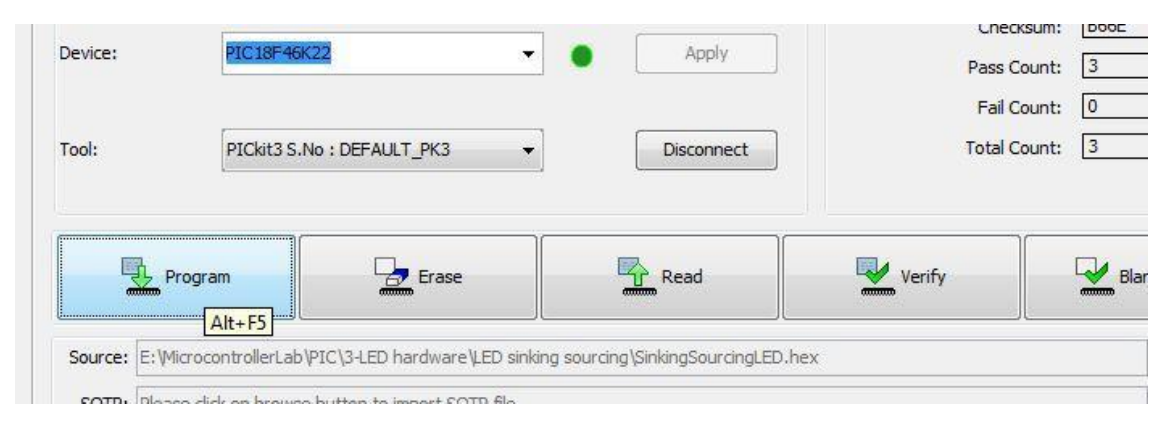

البرنامج (الكود) باستخدام احدى اللغات وهي ميكرو سي التي سوف ندرسها:

```
void main()
{
    TRISB0_bit = 0; //set pin B0 as output
    TRISB1_bit = 0; //set pin B1 as output
    ANSELB = 0; //set port as Digital
    do
    {
      LATB.B0 = 0;   //write the pin B0 with 0 value
      LATB.B1 = 1;   //write the pin B1 with 1 value
      Delay_ms(1000);  //Delay of 1 second
      LATB.B0 = 1;   //write the pin B0 with 1 value
      LATB.B1 = 0;   //write the pin B1 with 0 value
      Delay_ms(1000);  //Delay of 1 second
    }
    while(1);  //Keep the loop running
}
```

ان الكود الموجود في الصورة اعلاه لا يتجاوز الا بضعة اسطر.

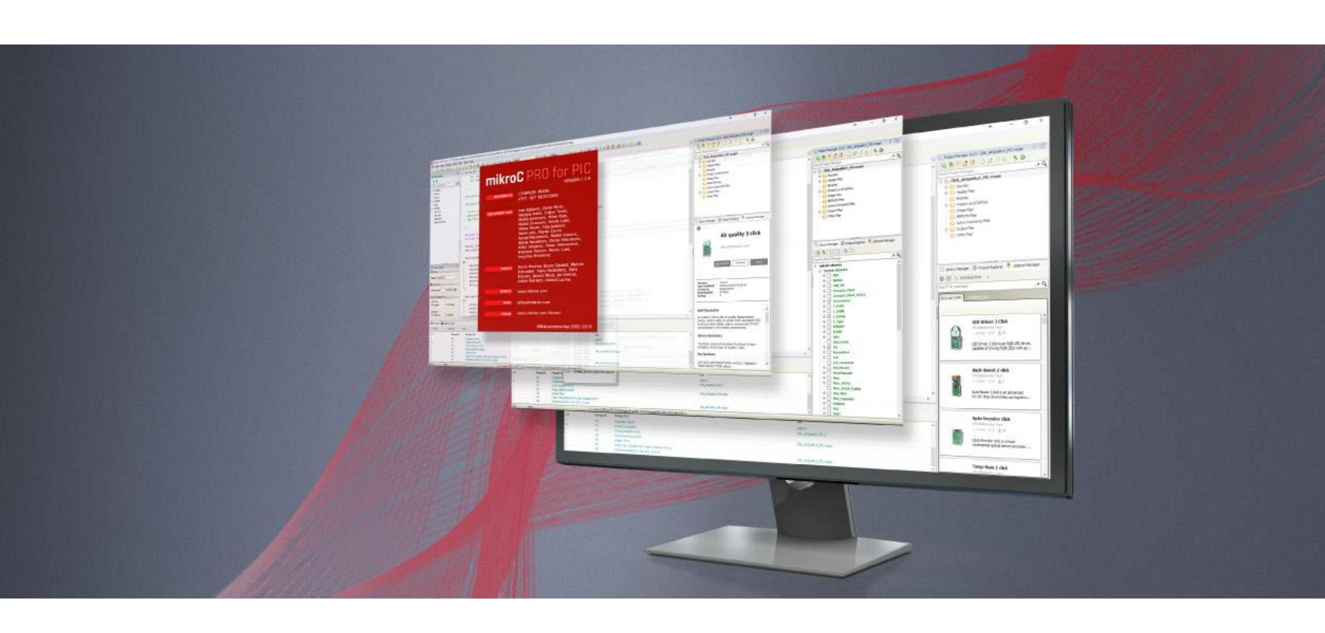
الشكل التالي يبين الواجهة الرئيسية لاحدى البرامج

ومن خلال البرنامج نقوم بتحميل الكود الى الشريحة




ان المصابيح الموجودة في الدائرة سوف تضيئ حسب اوامر الكود , كما ذكرت ليس من الضروري فهم الكود او البرنامج , هذا مثال توضيحي لطريقة التركيب البرمجة بشكل عام ويمكنك اختار اي مبرمجة تراها في السوق ولكن ابحث عن انواع جيدة لعدم تلف المتحكمات الالكترونية .

# تجهيز بيئة التطوير




يوجد العديد من لغات التي يمكن بها برمجة الميكرو كونترولر وتحديدا ال PIC وكذلك الامر في البرامج المستخدمة اي تلك البرامج التي تعمل عمل مترجم من الكود الذي نكتبه الى لغة الحاسوب 0,1 او نظام hex الذي تعرفنا عليه سابقا . لكن في هذا الكتاب سوف ندرس برمجة الميكروكونترولر باستخدام لغة ميكرو سي , ايضا باستخدام برنامج ميكروسي MICRO C . ان سبب اختيارنا لهذا البرنامج ان التعامل معه في غاية السهولة ويتوفر الكثير من الامثلة والمشاريع المتاحة عبر الانترنت .

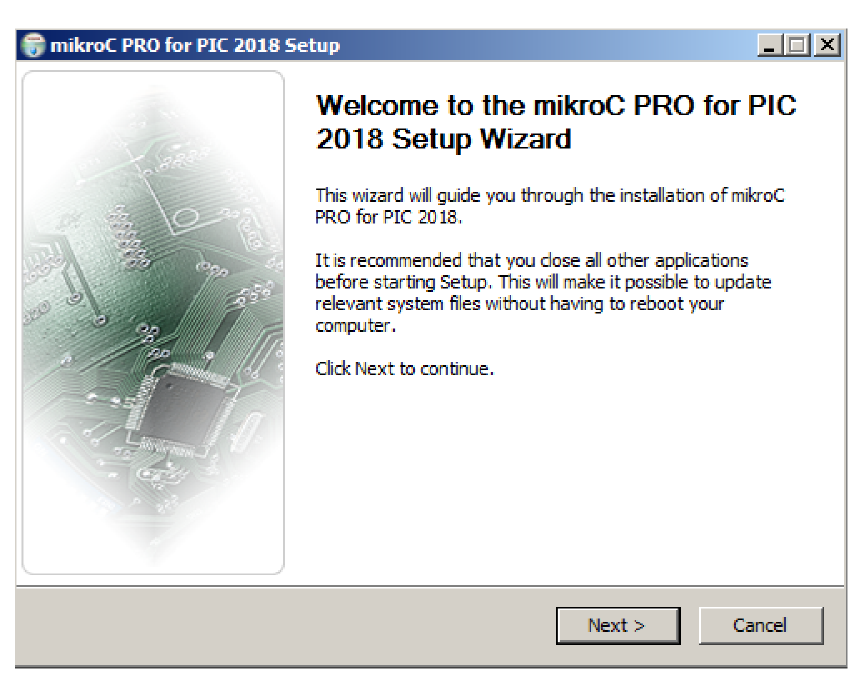
لتحميل البرنامج قم بالتوجه مباشرة الى الموقع الرسمي للشركة :

www.mikroe.com/mikroc-pic

او من خلال قرص محتويات الكتاب .

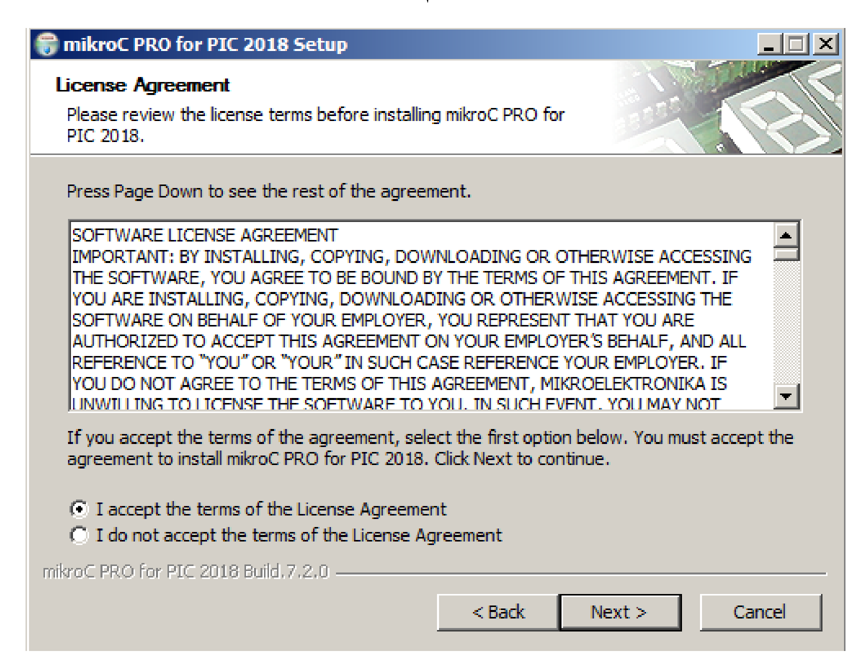
خطوات تنصيب البرنامج

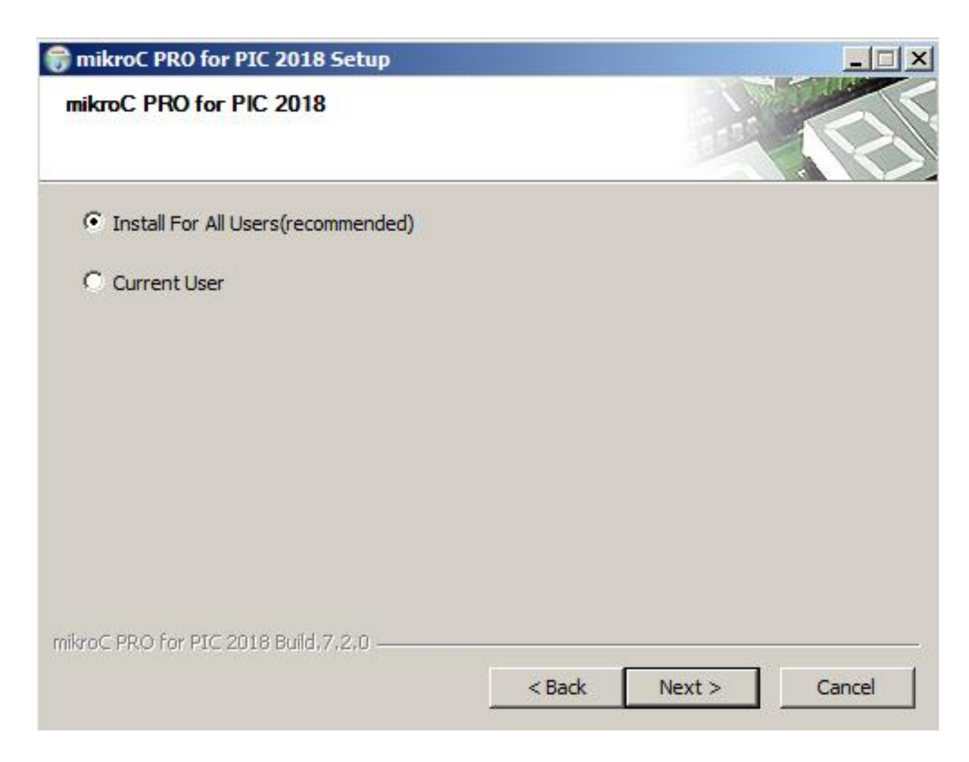
افتح ملف التحميل





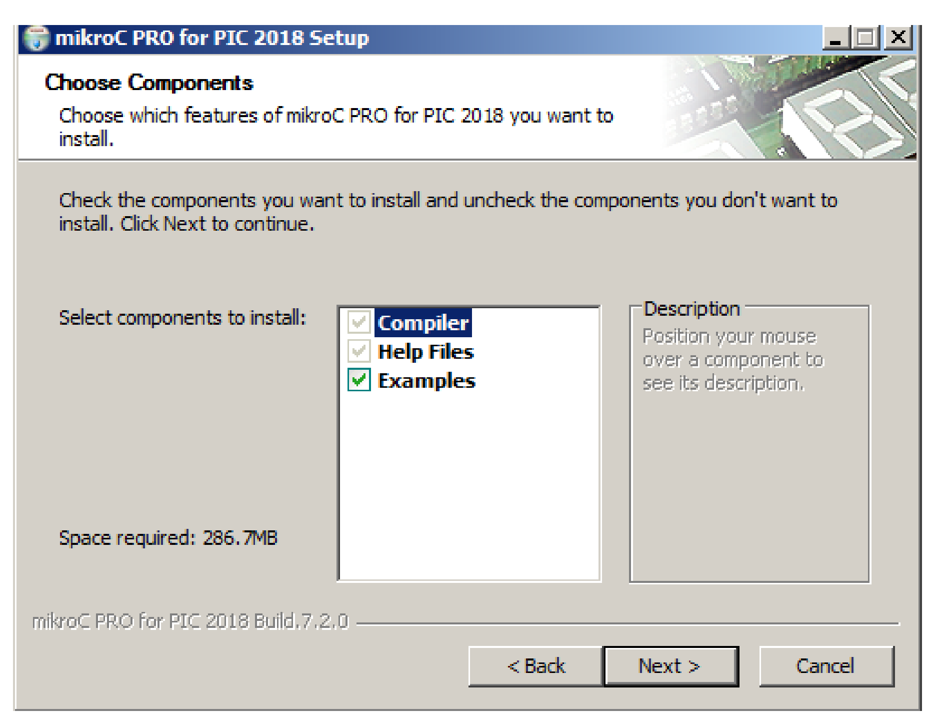
ثم



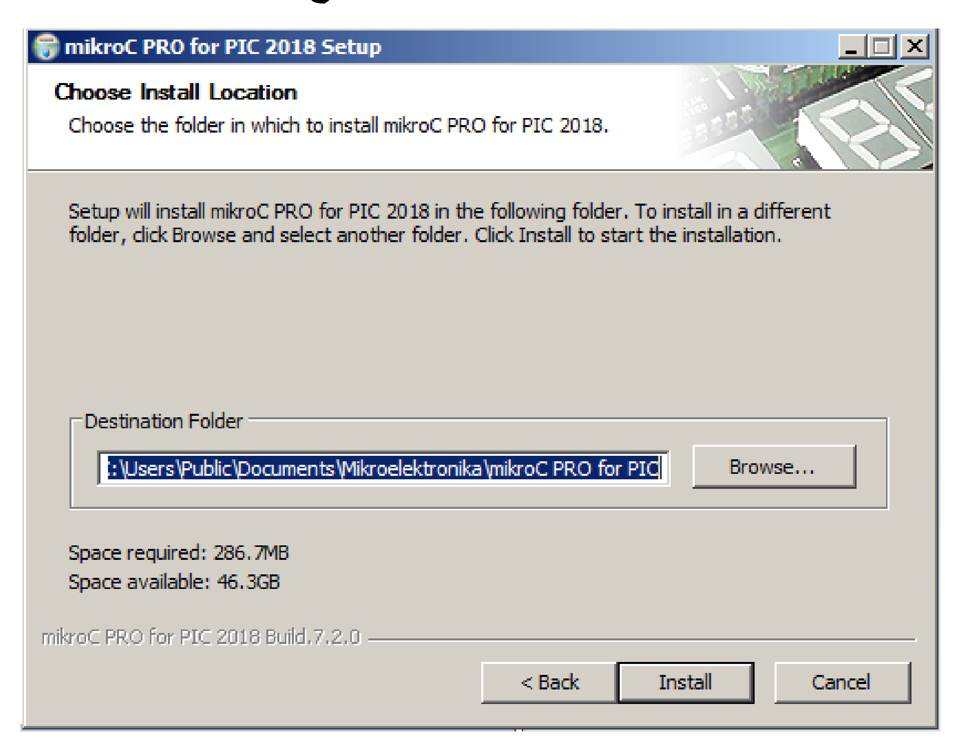

mikroC PRO for PIC 2018 Setup
mikroC PRO for PIC 2018
Install For All Users(recommended)
Current User
mikroC PRO for PIC 2018 Build.7.2.0
< Back
Next >
Cancel

ثم

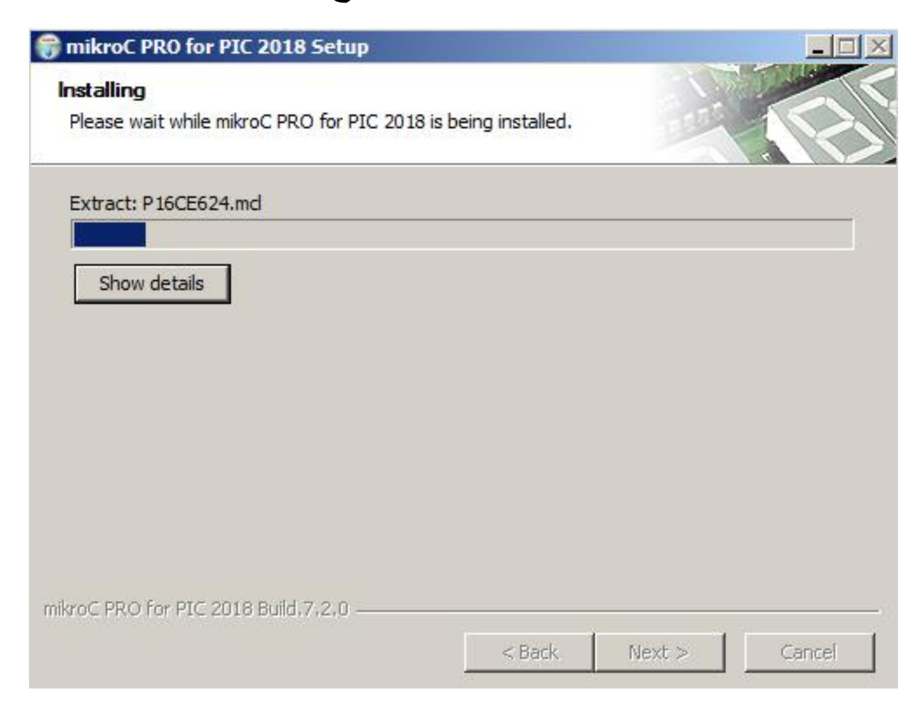



mikroC PRO for PIC 2018 Setup
Choose Components
Choose which features of mikroC PRO for PIC 2018 you want to install.
Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.
Select components to install:
Compiler
Help Files
Examples
Description
Position your mouse over a component to see its description.
Space required: 286.7MB
mikroC PRO for PIC 2018 Build.7.2.0
< Back
Next >
Cancel

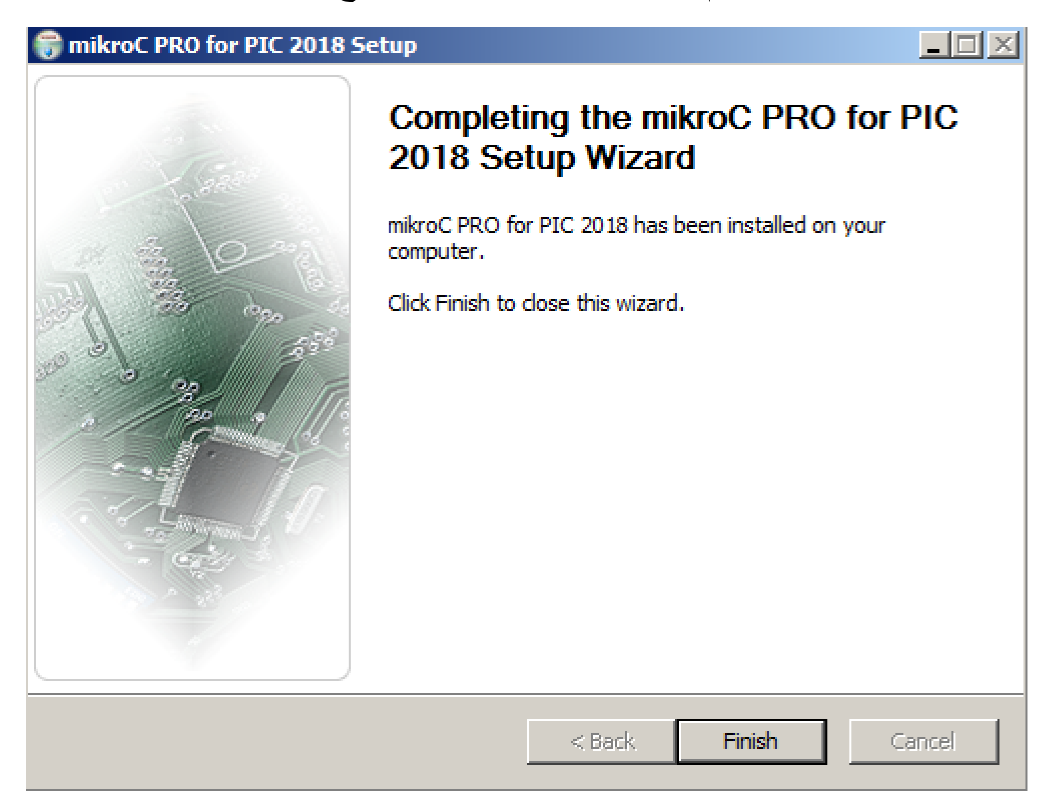
حدد مكان تنصيب البرنامج




سيبدأ تحميل البرنامج



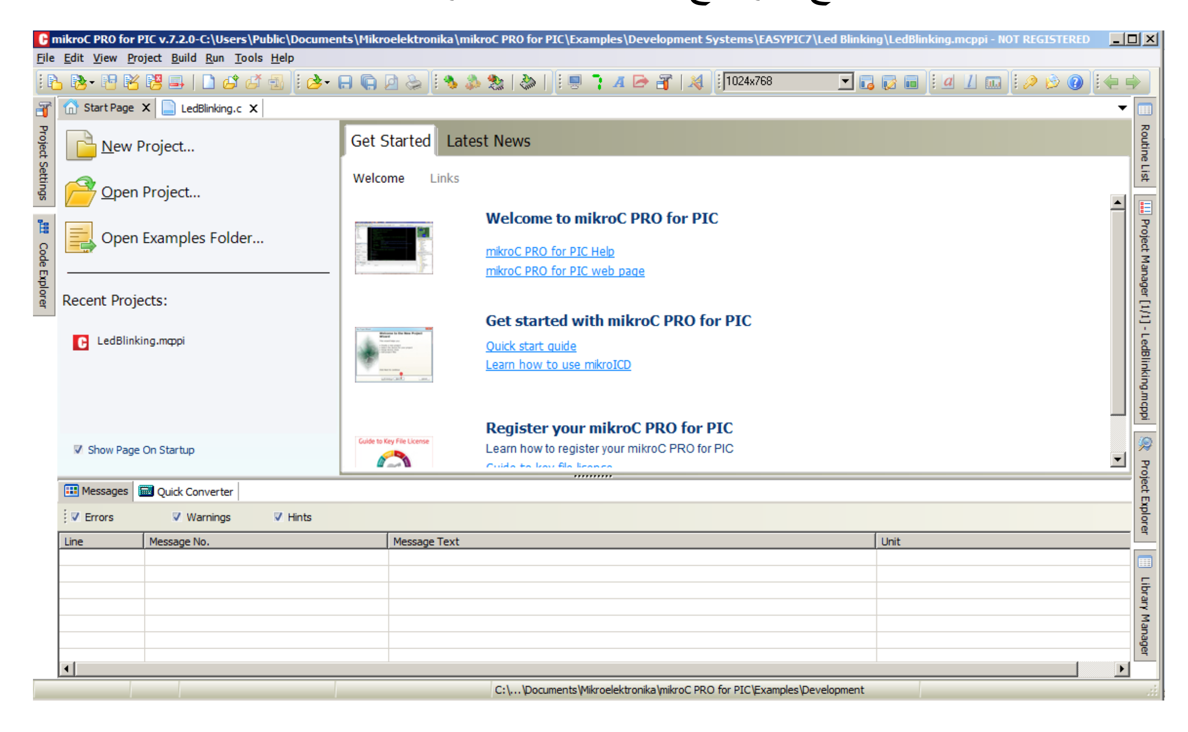
تم الانتهاء من تنصيب البرمج




نفتح البرنامج ثم تظهر لنا نافذة تحميل البرنامج



عند فتح البرنامج نلاحظ الشاشة الرئيسية التالية




يسمح لك mikroC PRO for PIC بالتطوير السريع للتطبيقات المعقدة ونشرها

اكتب التعليمات البرمجية المصدرية باستخدام Code Editor ( Code and Parameter Assistants, Code Folding, Syntax Highlighting, Auto Correct, Code Templates, and more.)

استخدام mikroC PRO المتوفر للمكتبات PIC لتسريع التطوير: الحصول على البيانات ، الذاكرة ، الشاشات ، التحويلات ، الاتصالات إلخ.

يمكن مراقبة بنية برنامجك ومتغيراته ووظائفه في Code Explorer

توليد الكود بلغة HEX القياسية المتوافقة مع جميع المبرمجين والمتحكمات والبرامج

استخدم أداة تصحيح الأخطاء In-Circuit Debugger في الوقت الحقيقي Real-Time لمراقبة تنفيذ البرنامج على مستوى العتاد الالكتروني .

فحص تدفق البرنامج وتصحيح منطق قابل للتنفيذ مع برنامج محاكاة البرمجيات المتكامل.

إنشاء ملف Common Object File Format= COFF لتصحيح البرامج والأجهزة تحت برنامج MPLAB Microchip.

تمكنك التعليقات النشطة Active Comments من جعل تعليقاتك حية وتفاعلية.

الحصول على تقارير ورسوم بيانية مفصَّلة: خريطة ذاكرة الوصول العشوائي و ROM، وإحصاءات الرموز ، وإدراج التجميع ، وشجرة الاتصال ، والمزيد ( RAM (and ROM map, code statistics, assembly listing, calling tree

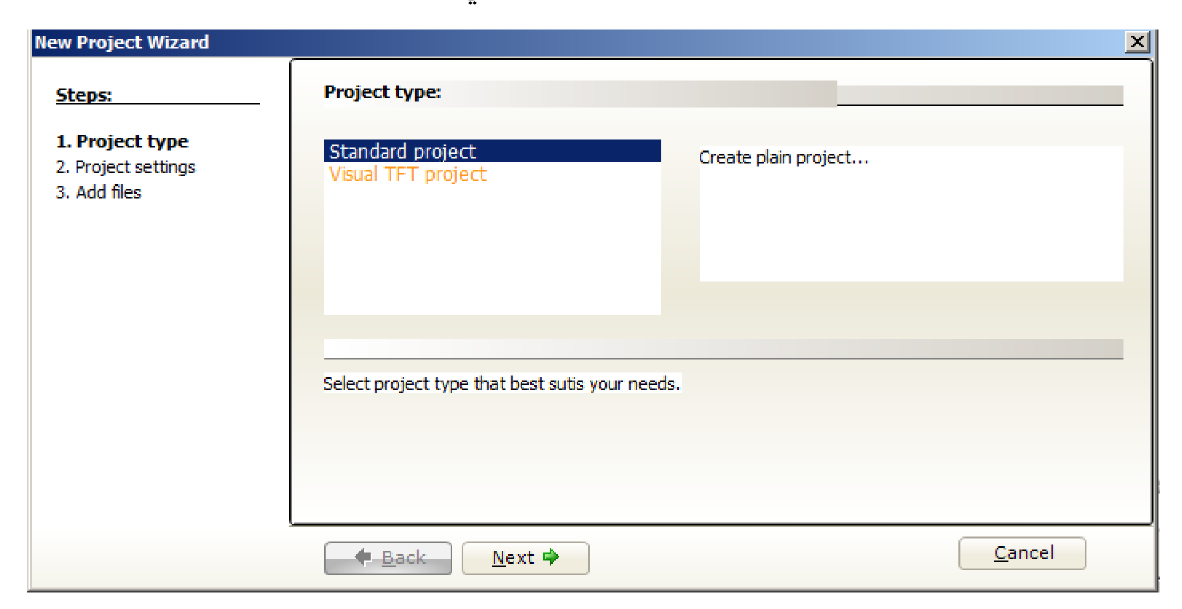
يوفر mikroC PRO for PIC الكثير من الأمثلة للتوسع والتطور في مشاريعك

حتى نبدا في تجهيز مكان كتابة الكود نتبع الخطوات التالية :

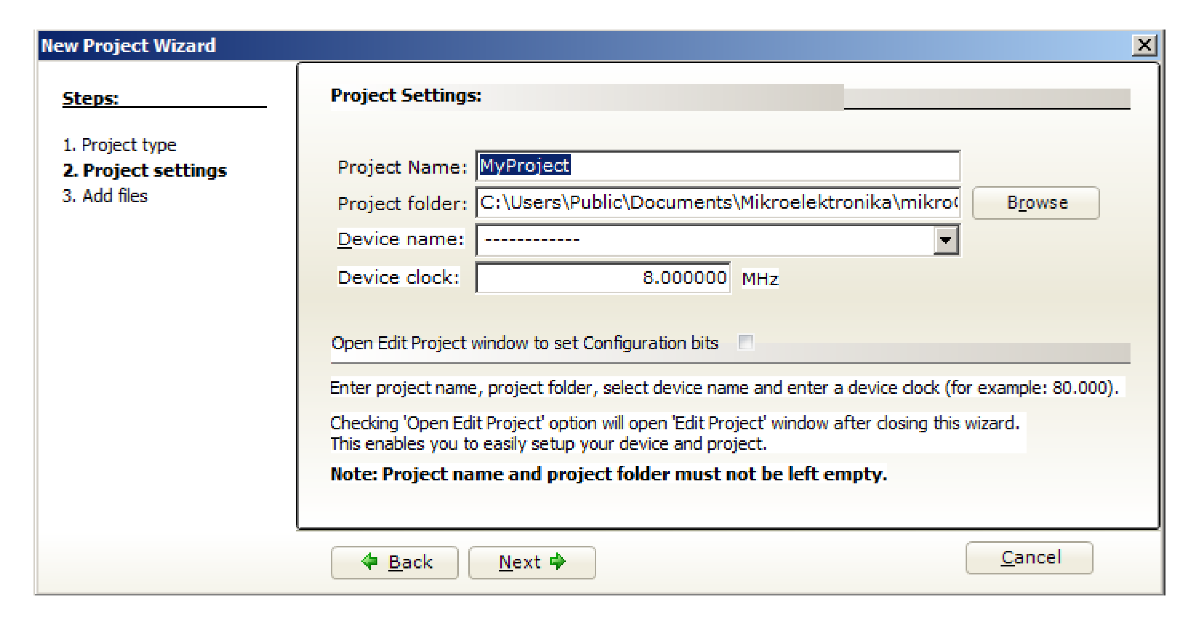
لبدأ مشروع جديد

Project -> New Project.

ستظهر النافذة التالي

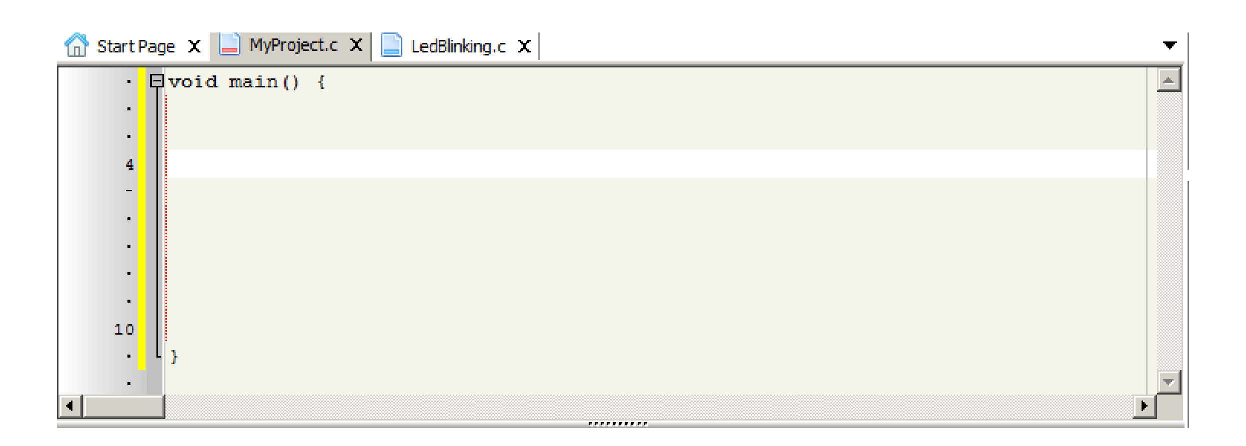



اضغط Next



نقوم بتعبئة اسم المشروع Project Name ومكان حفظ المشروع Project folder ونختار نوع ال PIC من Device name اما الخيار Device clock فسوف نشرحه لاحقًا .

بعد الضغط على Next وبعد تكوين الإعدادات الأولية ، يمكن كتابة الشفرة في فتح نافذة المحرر Code Editor الخاصة بلغة ميكرو سي

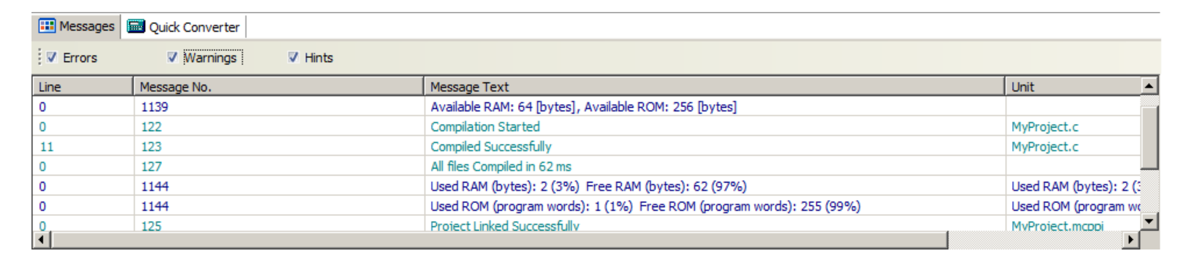



حسنا , ربما لاحظت ان البرنامج عند فتحه للمرة الاولى يقوم بفتح نافذة في المحرر باسم Led Blinking وهو مثال مثل Hello world في البرمجة كما تعلمنا سابقا في الجزء الثالث من هذه السلسلة .

نريد ان نتعلم ما يلي , قم بفتح التبويب الخاص ب Led Blinking واضغط

Project -> Build. او عن طريق Ctrl+F9

سيتم عرض الخطأ (الأخطاء) ، إن وجد ، في نافذة الرسائل في الأسفل كما يلي :




| Line | Message No. | Message Text | Unit |
|---|---|---|---|
| 0 | 1139 | Available RAM: 64 [bytes], Available ROM: 256 [bytes] | |
| 0 | 122 | Compilation Started | MyProject.c |
| 11 | 123 | Compiled Successfully | MyProject.c |
| 0 | 127 | All files Compiled in 62 ms | |
| 0 | 1144 | Used RAM (bytes): 2 (3%) Free RAM (bytes): 62 (97%) | Used RAM (bytes): 2 (3 |
| 0 | 1144 | Used ROM (program words): 1 (1%) Free ROM (program words): 255 (99%) | Used ROM (program wo |
| 0 | 125 | Project Linked Successfully | MyProject.mcppi |

:Debugging التصحيح

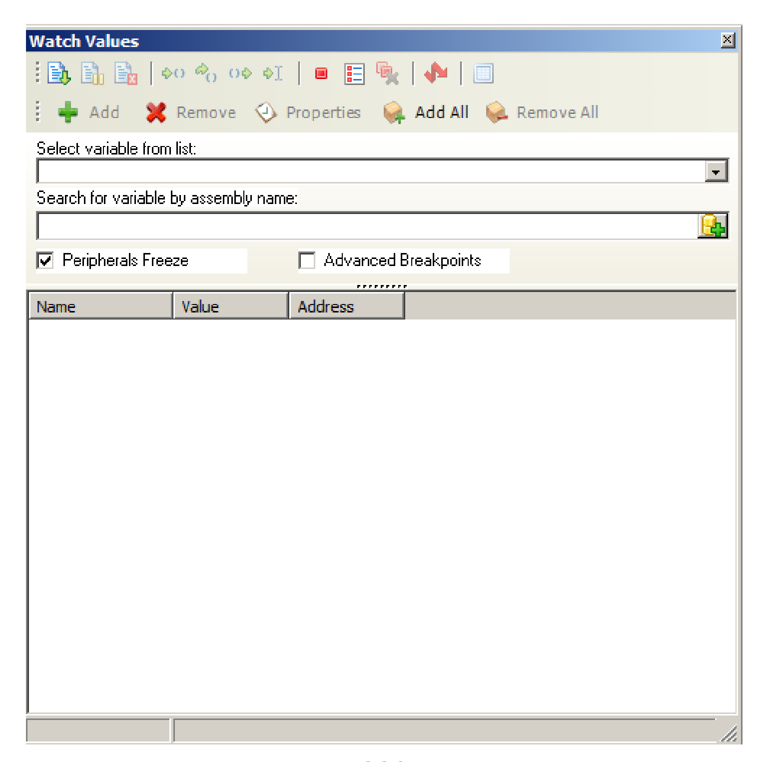
يمكن بدء تصحيح الكود باستخدام F9 أو من Run -> Start Debugging.

يمكن فتح نافذة المراقبة عن طريق

View -> Debug Window -> View Watch

يمكن إجراء تصحيح الأخطاء خطوة بخطوة عن طريق الضغط على F7 وسيتم عرض النتيجة المقابلة.

نافذة المراقبة :



عند إنشاء الكود ، يتم إنشاء ملف hex. في موقع الدليل المحدد الذي حددناه في بداية المشروع.

بعد الانتهاء من كتابة الكود وتحويله الى hex نستطيع حرقه (تحميله) الى ال PIC لتبدأ عملها عن طريق

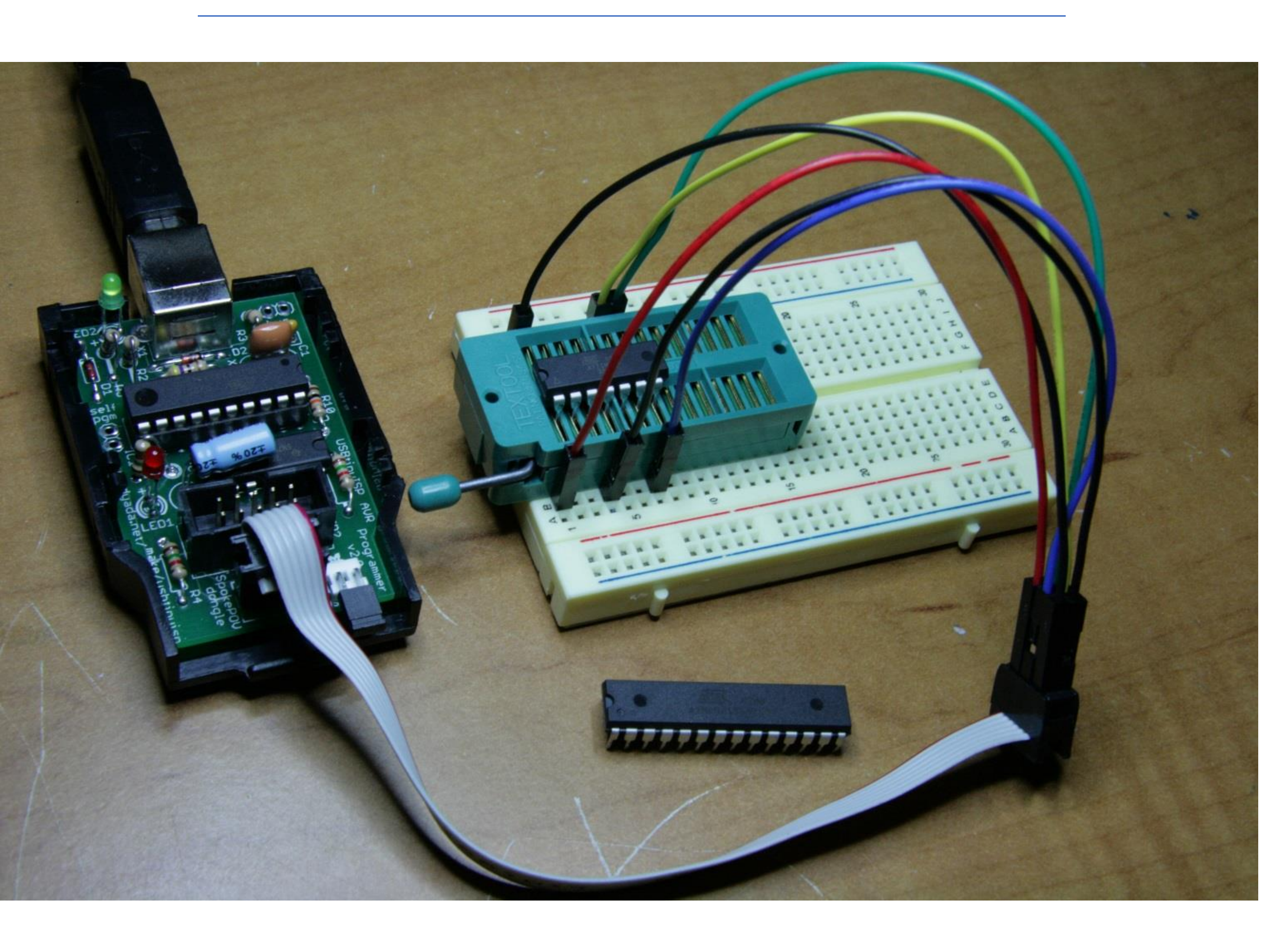
Build -> Build + Progam

او ctrl+ F11

البرنامج ليس بهذه البساطة فقط , يوجد الكثير جدا من التفاصيل فيما يتعلق بالاوامر والاختصارات وامكانيات البرنامج ولكن نحن نكتفي في هذا الكتاب بالقدر الذي يمكننا من كتابة كود (برنامج) بسيط للتحكم في تطبيقات يمكننا تطبيقها بسهولة .

يمكننا تلخيص العمل الذي سوف نقوم به بما تعلمنا بالخطوات التالية التي تمثل دورة تطوير المتحكمات PIC :

1) يتم كتابة الكود في نافذة المحرر Editor
2) يقوم المترجم Compiler / Assembler / Linker بتحويل الكود الى ملفات الدعم التي يفهمها المتحكم مثل hex
3) يتم تحميل الكودات البرمجية الى نافذة Simulator / Debugger
4) يتم تحليل الكود من الاخطاء ان وجد وتحميله الى ال PIC



حسنا بعد ان تعرفنا على انواع الميكرو كونترولر يمكنك اختيار نوع المتحكم PIC الذي تراه مناسبا حسب المواصفات التي شرحناها سابقا وايضا كذلك الامر بالنسبة للمتحكمة (قطعة البرمجة) التي بها ستبرمج ال PIC , وايضا لك الحرية في اختيار البرنامج الذي ستعمل به , في هذا الدرس سوف نقوم بدراسة لغة البرمجة ميكرو سي لكن قبل في هذا الدرس سوف نشرح قطعة مهمة في الميكروكونترلر وهي الكريستالة او المذبذب.

# كريستال Crystal




يحتاج الميكروكونترولر إلى ساعة للعمل. تحتوي معظم المتحكمات الدقيقة على مذبذب RC داخلي يخلق إشارة على مدار الساعة , مثل ATmega32U2.

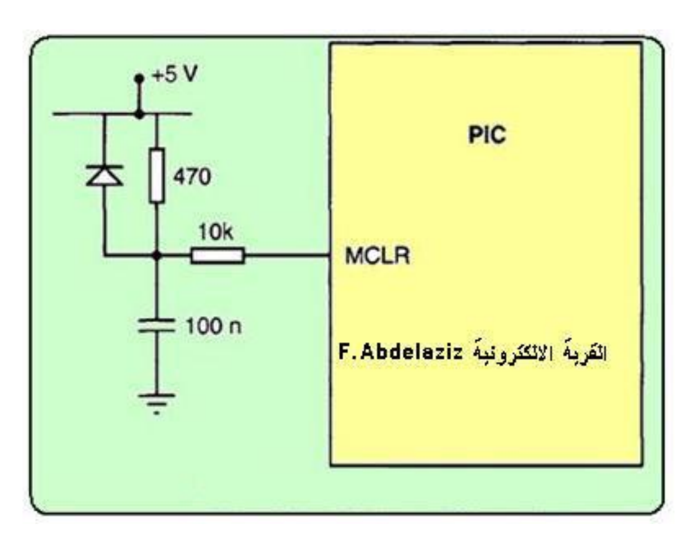
غالبا ما يستخدم مذبذب الكريستالة نظرا لدقته وسرعته حيث أن مذبذب RC والذى يعتمد على الجهد الداخل الى المقاومة والمكثف والذى يولد بما يسمى resonant frequency يعيبه أنه ليس دقيق حيث انه عند توصيل مصدر التغذية له لا يعمل بالسرعة المرجوة فهو يأخذ وقت يسمى وقت بداية المذبذب حتى يعمل حيث تتأثر نبضاته بمعدل تغير الجهد وشدة التيار. والشكل اعلاه يبين شكل الكرستالة المستخدمة كمذبذب , وتختلف الكريستالات حسب الشكل والتردد الذي تعطيه .

والكريستالة ليس لها قطبية فيتم توصيل أى من طرفيها على الطرفان , OSC1 OSC2 ويراعى توصيل 2 مكثف سيراميكي بسعة 10 pf واحيانا 22 pf على كل طرف من أطرافها وبين الأرضي حيث أنها تستخدم لمنع الضجيج. ويمكن الاستغناء عن المذبذب الكريستالي بشرط الحصول على الذبذبات من أي مصدر خارجي كمولد للذبذبات

بشكل عام نحتاج إلى دائرتين خارجيتين احيانا للتعامل مع ال PIC : دائرة التصفير reset ودائرة المذبذب oscillator

دائرة التصفيرreset

يتم تحقيق التصفير بتوصيل مقاومة رفع توصل من المدخل MCLR إلى جهد التغذية الموجب . أحيانا يرتفع الجهد ببطىء ولا تعمل وظيفة التصفير العادية البسيطة لذلك يجب استخدام دائرة تصفير خارجية كالمبينة بالشكل .

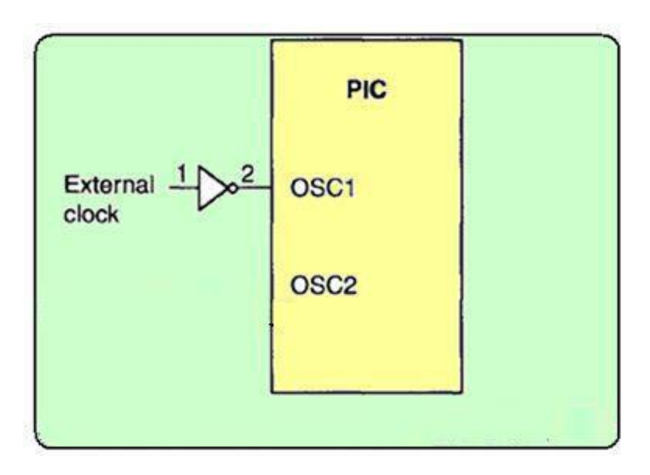



دائرة المذبذب oscillator

يحتوى الميكروكونترولر في بناءه الداخلي على دوائر مذبذبات والتى يمكن أن تعمل فى أحد خمسة أنظمة:

- نظام كريستال بقدرة منخفضة. LP
- نظام كريستال / دائرة رنين. XT
- نظام كريستال / دائرة رنين بسرعة عالية. HS
- نظام مقاومة – مكثف. RC
- نظام داخلى كامل بدون أى مكونات خارجية.

فى الأنظمة LP , XT , HS يمكن توصيل مذبذب خارجى إلى طرف الدخل OSC1 كما فى الشكل التالي :



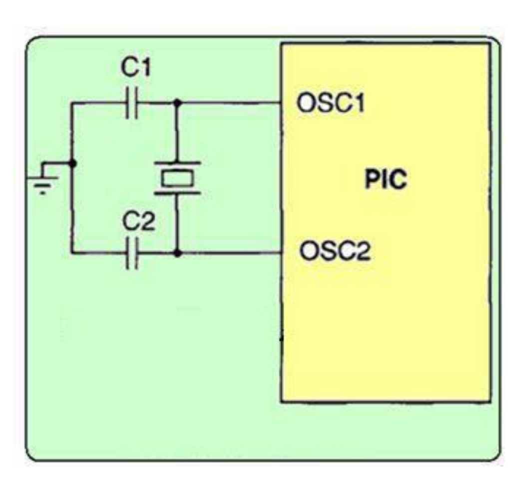

النظام : LP مصمم للعمل بالسرعة العادية وبقدرة منخفضة حيث تستهلك الدوائر الداخلية لنظام LP تيار أقل وهذا النظام مناسب للعمل على التردد المنخفض.

النظام : XT مصمم كحل للتوفيق بين العمل عند التردد المرتفع والاستهلاك المناسب للقدرة . ينصح باستخدام هذا النظام للعمل حتى 4. MHz

النظام : HS ويعرف أيضا بنظام السرعة المرتفعة ويعطى هذا النظام استجابة عند أعلى الترددات . استهلاك التيار يكون أيضا الأعلى من الانظمة الأخرى . ينصح باستخدام نظام HS للعمل حتى MHz20 .

استخدام نظام الكريستال :

كما هو موضح بالشكل , فى هذا النظام يتم توصيل كريستال ومكثفين خارجيا إلى مداخل الميكروكونترولر OSC1 و OSC2 . يجب اختيار المكثفات كما فى الجدول المرفق . على سبيل المثال عند استخدام كريستال بتردد MHz4 نستخدم مكثفان بالقيمة pF22 .

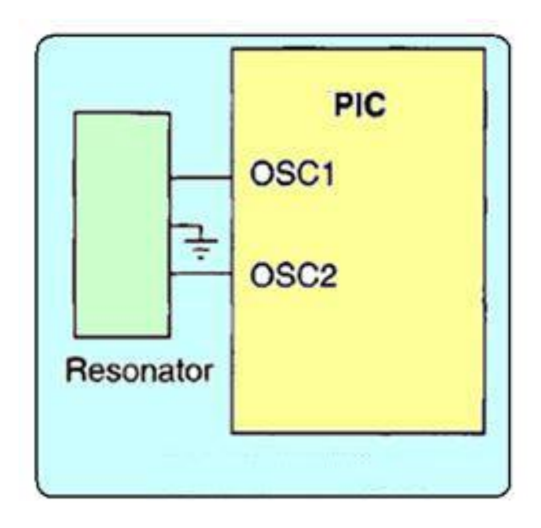



جدول اختيار المكثفات عند استخدام الكريستال

| Mode | Frequency | C1, C2 |
|---|---|---|
| LP | 32 kHz | 68-100 pF |
| LP | 200 kHz | 15-33 pF |
| XT | 100 kHz | 100-150 pF |
| XT | 2 MHz | 15-33 pF |
| XT | 4 MHz | 15-33 pF |
| HS | 4 MHz | 15-33 pF |
| HS | 10 MHz | 15-33 pF |

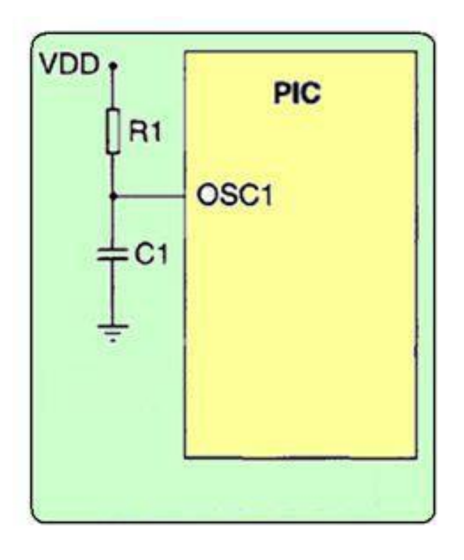
استخدام دائرة الرنين Resonator

دائرة الرنين متوفرة بتردد من 4 MHz إلى حوالي 8 , MHz وهى ليست فى دقة الكريستال . الشكل يبين طريقة التوصيل.




استخدام نظام المذبذب: RC

فى التطبيقات التى لا تحتاج إلى دقة فى التوقيت يمكننا توصيل مقاومة ومكثف خارجيا إلى مدخل الميكروكونترولر OSC1 كما فى الشكل . يعتمد تردد المذبذب على كل من : قيم كل من المقاومة والمكثف (انظرالجدول المرفق) وجهد التغذية و درجة الحرارة . فى معظم التطبيقات نستخدم مقاومة k5 مع مكثف pF20 لكى نحصل على تردد بحوالى MHz4 وهى قيمة مقبولة .



جدول اختيار قيمة المقاومة والمكثف للمذبذب RC

| C1 | R1 | Frequency |
|---|---|---|
| 20 pF | 5k | 4.61 MHz |
| | 10k | 2.66 MHz |
| | 100k | 311 kHz |
| 100 pF | 5k | 1.34 MHz |
| | 10k | 756 kHz |
| | 100k | 82.8 kHz |
| 300 pF | 5k | 428 kHz |
| | 10k | 243 kHz |
| | 100k | 26.2 kHz |

بعض الميكروكونترولر PIC تحتوى داخليا على دائرة مذبذب كامل ولا تحتاج أى مكونات خارجية .عادة ما يكون المذبذب الداخلى بتردد MHz4 ويتم اختياره عند البرمجة . يجب اختيار المذبذب الداخلي عند التطبيقات رخيصة التكاليف .

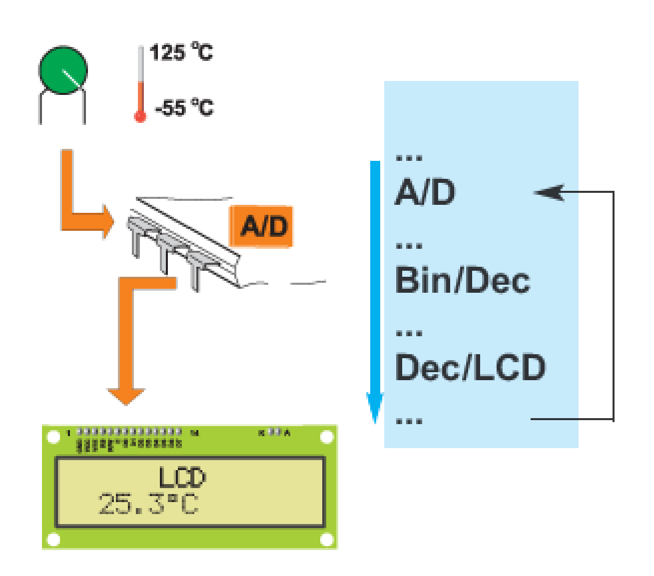
حسنا لنبدأ رحلة تعلم الميكرو سي !

mikroC PRO for PIC

VERSION 7.2.0

©Mikroelektronika 2002-2019

الفكرة الرئيسية لكتابة البرنامج باللغة C هي تجزيئ الادوات إلى عدة أجزاء صغيرة لبرمجتها. افترض أنه من الضروري كتابة برنامج من أجل وحدة التحكم الدقيقة التي ستقوم بقياس درجة الحرارة وعرض النتائج على شاشة LCD. يتم إجراء عملية القياس بواسطة جهاز استشعار يحول درجة الحرارة إلى جهد كهربائي. يستخدم الميكروكونترولر محوله A / D لتحويل هذا الجهد (القيمة التناظرية) إلى (قيمة رقمية) والذي يتم إرساله بعد ذلك إلى شاشة LCD عبر عدة موصلات. وبناءً عليه ، يتم تقسيم البرنامج إلى أربعة أجزاء يجب عليك القيام بها وفقًا للترتيب التالي:

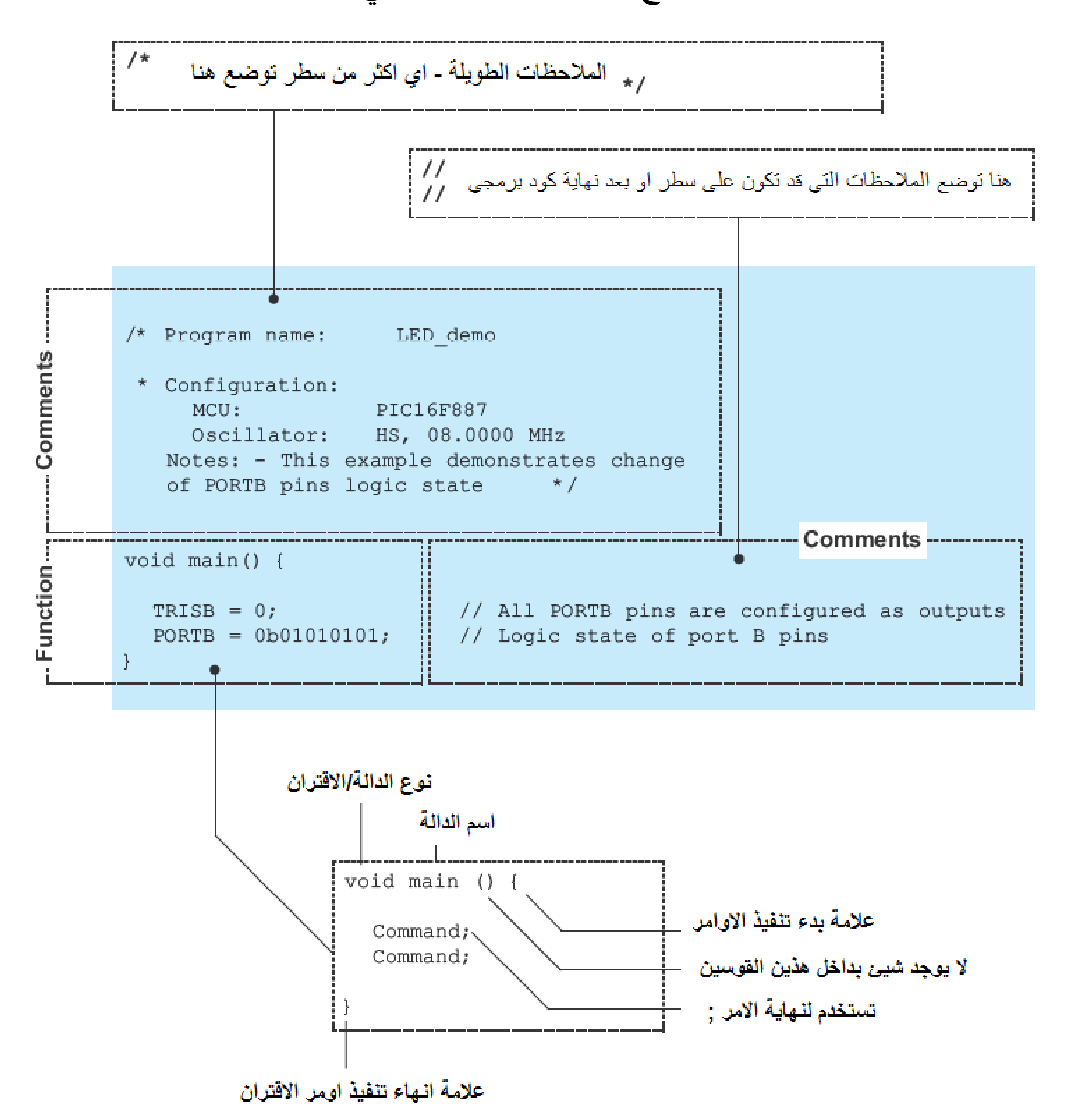



1) تنشيط وتعيين محول.A / D
2) قياس القيمة التماثلية
3) احسب درجة الحرارة
4) إرسال البيانات في النموذج المناسب لشاشة LCD.

كما رأينا ، تمكنك لغات البرمجة ذات المستوى العالي مثل C من حل هذه المشكلة بسهولة عن طريق كتابة أربع وظائف ليتم تنفيذها بشكل دوري ومرارًا وتكرارًا.

- التعليقات او الملاحظات : تعتبر التعليقات جزءًا من البرنامج المستخدم لتوضيح تشغيل البرنامج أو تقديم المزيد من المعلومات عنه. يتم تجاهل التعليقات ولا يتم تحويلها إلى كود قابل للتنفيذ بواسطة المحول البرمجي compiler. وببساطة ، يستطيع المحول البرمجي التعرف على الأحرف الخاصة المستخدمة لتعيين مكان بدء التعليقات وإنهائها وتجاهل النص بالكامل أثناء التجميع. هناك نوعان من هذه التعليقات , يعيّن أحد التعليقات الطويلة التي تمدّ العديد من خطوط البرامج ، بينما يعيّن الآخر تعليقات قصيرة تتناول سطرًا واحدًا. على الرغم من أن التعليقات لا يمكن أن تؤثر على تنفيذ البرنامج ، إلا أنها مهمة مثل أي جزء آخر من البرنامج ، و هنا السبب يمكن دائمًا تحسين برنامج مكتوب وتعديله وترقيته وتبسيطه , يتم تنفيذه دائمًا تقريبًا اذا كانت التعليقات موجودة وموضحة للمستخدم!. من دون تعليقات ، فإن محاولة فهم أبسط البرامج هي إضاعة للوقت.

لنوضح ما سبق بالهيكل التالي :



- أنواع البيانات باللغة C

هناك عدة أنواع من البيانات التي يمكن استخدامها في لغة البرمجة C. يوضح الجدول أدناه القيم التي يمكن أن تحتويها هذه البيانات عند استخدامها في :

| نوع البيانات | الوصف | الحجم (عدد البيتات) | مدى القيم |
|---|---|---|---|
| char | Character | 8 | 0 to 255 |
| int | Integer | 16 | -32768 to 32767 |
| float | Floating point | 32 | $\pm1.17549435082 \cdot 10^{-38}$ to $\pm6.80564774407 \cdot 10^{38}$ |
| double | Double precision floating point | 32 | from $\pm1.17549435082 \cdot 10^{-38}$ to $\pm6.80564774407 \cdot 10^{38}$ |

بإضافة الامر prefix (qualificator) اي المؤهل إلى أي نوع بيانات ، يتغير نطاق القيم المحتملة بالإضافة إلى عدد وحدات الذاكرة المطلوبة :

| نوع البيانات | نوع البيانات مع prefix | الحجم (عدد البيتات) | مدى القيم |
|---|---|---|---|
| char | signed char | 8 | -128 to 128 |
| int | unsigned int | 16 | 0 to 65535 |
| | short int | 8 | 0 to 255 |
| | signed short int | 8 | -128 to 127 |
| | long int | 32 | 0 to 4294967295 |

| | | | |
|---|---|---|---|
| | signed long int | 32 | -2147483648 to 2147483647 |

- المتغيرات

يسمى أي رقم يغير قيمته أثناء تشغيل البرنامج متغيرًا. ببساطة ، إذا كان البرنامج يضيف رقمين number1 و number2 ، فمن الضروري أن تكون لديك قيمة لتمثل ما نسميه في الحياة اليومية المبلغ. في هذه الحالة ، يكون number1 و number2 متغيرين.

اعلان المتغيرات Declaring Variables

1) يمكن أن يشتمل اسم المتغير على أي من الحروف الأبجدية (A-Z (a-z والأرقام من 0 إلى 9 ورمز الشرطة السفلية '_'. المحول البرمجي compiler حساس لحالة الأحرف ويفرق بين الاحرف الكبيرة والأحرف الصغيرة. تحتوي أسماء الدالة والمتغير عادةً على أحرف صغيرة ، بينما تحتوي أسماء الثوابت على أحرف كبيرة.
2) يجب ألا تبدأ أسماء المتغيرات برقم.
3) لا يمكن استخدام بعض الأسماء كأسماء متغيرة كما سبق أن تم استخدامها من قبل المجمع نفسه. تسمى هذه الأسماء الكلمات الأساسية او الكلمات المحجوزة KEYWORDS. مترجم mikroC يتعرف على ما مجموعه 33 كلمة موضحات في الجدول ادناه :

| MIKROC - KEYWORDS | | | | |
|---|---|---|---|---|
| absolute | data | if | return | typedef |
| asm | default | inline | rx | typeid |

| | | | | |
|---|---|---|---|---|
| at | delete | int | sfr | typename |
| auto | do | io | short | union |
| bit | double | long | signed | unsigned |
| bool | else | mutable | sizeof | using |
| break | enum | namespace | static | virtual |
| case | explicit | operator | struct | void |
| catch | extern | org | switch | volatile |
| char | false | pascal | template | while |
| class | float | private | this | |
| code | for | protected | throw | |
| const | friend | public | true | |
| continue | goto | register | try | |

## المؤشرات Pointers

المؤشر هو نوع خاص من المتغيرات يحمل عنوان متغيرات الحرف. بمعنى آخر ، يشير المؤشر إلى "متغير آخر". يتم التصريح (ادراجها) على النحو التالي:

```
type_of_variable *pointer_name;
```

لتعيين عنوان متغير إلى مؤشر ، من الضروري استخدام الحرف '=' وكتابة اسم المتغير مسبوقًا بالحرف '&'.

في المثال التالي ، يتم الإعلان عن إرسال متعدد multiplex المؤشر وتعيين عنوان أول من ثمانية شاشات LED

```
unsigned int *multiplex; // Declare name and type of pointer multiplex
multiplex = &display1;   // Pointer multiplex is assigned the address of
                         // variable display1
```

لتغيير قيمة المؤشر للمتغير ، يكفي كتابة الحرف "*" أمام المؤشر وتعيين قيمة جديدة له كالتالي :

```
*multiplex = 6; // Variable display1 is assigned the number 6
```

وبالمثل ، من أجل قراءة قيمة المؤشر للمتغير ، يكفي كتابة:

```
temp = *multiplex; // The value of variable display1 is copied to temp
```

## تغيير البتات الفردية individual bits

هناك عدة طرق لتغيير جزء واحد فقط من المتغير (one bit of a variable). أبسط طريقة هي تحديد اسم السجل register name أو موضع البت bit's position أو الاسم والحالة المرغوبة desired state :

لحذف البتات الموجودة في RD3 واذا اردنا تحديد TC مخرج واحد بيت نكتب الصيغة التالية :

```
(PORTD.F3 = 0) ;    // Clear the RD3 bit
...
(PORTC.RELAY = 1) ; // Set the PORTC output bit (previously named RELAY)
                    // RELAY must be defined as constant
```

يجب إعلان كل متغير قبل استخدامه لأول مرة في البرنامج. بما أن المتغيرات مخزنة في ذاكرة RAM ، فمن الضروري حجز مساحة لها (واحد أو اثنين أو أكثر من البايت).

أنت تعرف نوع البيانات التي تكتبها أو تتوقعها نتيجة لعملية ما ، بينما لا يعرف المترجم ذلك. لا تنسَ أن البرنامج يتعامل مع المتغيرات التي عيّنت لها النوع وكل نوع له حدوده وخصائص . يتعرف المحول البرمجي على أنها سجلات ذاكرة .RAM عادة ما يتم تعيين أنواع المتغيرات في بداية البرنامج.

وهذا مثال لتعريف متغير يحمل اسم gate1 :

```
unsigned int gate1; // Declare name and type of variable gate1
```

وهذه بعض الامثلة الجيدة , في السطر الثاني قمنا بتعريف متغيرين في نفس السطر وفي السطر الثالث قمنا بتعيين قيمة للمتغير :

```
unsigned int gate1;    // Declare type and name of the variable
signed int start, sum; // Declare type and name of other two variables
gate1 = 20;            // Assign variable gate1 an initial value
```

لكن يمكن تسمية المتغيرات في سطر واحد كالتالي :

```
unsigned int gate1=20; // Declare type, name and value of variable
```

إذا كان هناك العديد من المتغيرات التي تم تعيينها بنفس القيمة الأولية ، يمكن تبسيط العملية حتى:

```
unsigned int gate1=gate2=gate3=20;
signed int start=sm=0;
```

نوع المتغير الغير معرف بعلامة "+" أو "-" اي بشكل افتراضي. على سبيل المثال ، يمكن كتابة char بدلاً من signed char . في هذه الحالة ، يأخذ المترجم قيمًا إيجابية للمتغيرات

إذا نسيت ، أن تعلن عن نوع متغير ، فإن المترجم سيعتبرها تلقائيًا signed integer

وهو يعني أن مثل هذا المتغير سيشغل بايت ذاكرة ويكون لهما قيم في نطاق 32768- الى 32767+

## الثوابت CONSTANTS

الثابت هو رقم أو حرف له قيمة ثابتة لا يمكن تغييرها أثناء تنفيذ البرنامج. على عكس المتغيرات ، يتم تخزين الثوابت في ذاكرة برنامج الفلاش flash program memory من المتحكم الدقيق لغرض توفير مساحة كبيرة من ذاكرة الوصول العشوائي. يتعرف المحول البرمجي عليها بواسطة الدالة / الوظيفة const.

## INTEGER CONSTANTS

يمكن أن تكون الثوابت الصحيحة عشريًا أو سداسيًا عشريًا أو ثمانيًا أو ثنائيًا. يتعرف المحول البرمجي على تنسيقه على أساس البادئة المضافة/الدالة prefix. إذا كان الرقم لا يحتوي على بادئة ، فسيتم اعتباره عشريًا افتراضيًا decimal by default.

يتم التعرف على نوع الثابت تلقائيًا من خلال حجمه ,في المثال التالي سيتم اعتبار MINIMUM الثابت تلقائيًا عددًا صحيحًا موقّعًا وتخزينه ضمن وحدتي بايت من ذاكرة الفلاش (16 بت)

```
const MINIMUM = -100; // Declare constant MINIMUM
```

انظر الجدول التالي :

| FORMAT | PREFIX | EXAMPLE |
|---|---|---|
| Decimal | - | const MAX = 100 |
| Hexadecimal | 0x or 0X | const MAX = 0xFF |
| Octal | 0 | const MAX = 016 |

| | | |
|---|---|---|
| Binary | 0b or 0B | const MAX = 0b11011101 |

## FLOATING POINT CONSTANTS

يتكون Float من جزء int، نقطة ، جزء كسري و e او E اختياري وهذا المثال يوضح طريقة استخدامه :

```
const T_MAX = 32.60; // Declare temperature T_MAX
const T_MAX = 3.260E1; // Declare the same constant T_MAX
```

في كلا المثالين ، يتم تعريف ثابت يسمى T_MAX لديه قيمة كسورية 32.60 إنه يمكّن البرنامج من مقارنة درجة الحرارة المقاسة إلى الثابت ذي المعنى بدلاً من الأرقام التي تمثله.

## CHARACTER CONSTANTS (ASCII CHARACTERS)

ثابت الأحرف هو حرف محاط بعلامات اقتباس مفردة. في المثال التالي ، يتم تعريف ثابت يسمى I_CLASS كحرف A ، بينما يتم تعريف ثابت يسمى II_CLASS كحرف B.

```
const I_CLASS = 'A';  // Declare constant I_CLASS
const II_CLASS = 'B'; // Declare constant II_CLASS
```

عند التعريف هذه الطريقة ، يؤدي تنفيذ الأوامر بإرسال ثابتي I_CLASS و II_CLASSإلى شاشة LCD ، إلى عرض الحرفين A و B ، على التوالي.

# STRING CONSTANTS

يسمى الثابت الذي يتكون من تسلسل من الأحرف string يتم تضمين ثوابت String ضمن علامات اقتباس مزدوجة.

```
const Message_1 = "Press the START button"; // Message 1 for LCD
const Message_2 = "Press the RIGHT button"; // Message 2 for LCD
const Message_3 = "Press the LEFT button";  // Message 3 for LCD
```

في هذا المثال ، سيؤدي إرسال ثابت Message_1 إلى شاشة LCD إلى ظهور الرسالة Press the START button

# ENUMERATED CONSTANTS

هي نوع خاص من الثوابت الصحيحة التي تجعل البرنامج أكثر شمولاً وأسهل في تتبعه عن طريق تعيين عناصر الأرقام الترتيبية. في المثال التالي ، يتم تعيين القيمة الأولى للعنصر الأول في الأقواس المتعرجة تلقائيًا ، بينما يتم تعيين القيمة الثانية في القيمة 1 ، والثالث القيمة 2 وما إلى ذلك.

```
enum MOTORS {UP, DOWN, LEFT, RIGHT}; // Declare constant MOTORS
```

في كل تكرار لكلمات " LEFT" ، "RIGHT " ، "UP"و في البرنامج ، سيحل المحلل محلهم بالأرقام المناسبة (0-3) بشكل منتظم ، إذا كان المنفذ 0 و 1 و 2 و 3 متصلاً بمحركات مما يجعل شيء ما يرتفع إلى الأعلى والأسفل واليمين واليسار ، فإن الأمر الخاص بتشغيل المحرك 'RIGHT' المتصل بالبتة 3 من المنفذ B يبدو كما يلي:

```
PORTB.RIGHT = 1; // set the PORTB bit 3 connected to the motor 'RIGHT'
```

Operator هو رمز يدل على العمليات الحسابية أو المنطقية أو بعض العمليات الأخرى. هناك أكثر من 40 عملية متوفرة في لغة C ، ولكن على الأكثر 10-15 منها تستخدم في البرمجة يتم تنفيذ كل عملية على واحد أو أكثر من المعاملات التي يمكن أن تكون متغيرات أو ثوابت. إلى جانب ذلك ، تتميز كل عملية بتنفيذ الأولوية الخاصة بها

## ARITHMETIC OPERATORS

تستخدم Arithmetic operators في العمليات الحسابية وتعود غالبًا إلى نتائج موجبة. على عكس العمليات غير العادية التي يتم تنفيذها على مُعامل واحد ، يتم تنفيذ العمليات الثنائية على اثنين من المعاملات. بمعنى آخر ، مطلوب رقمين لتنفيذ عملية ثنائية. على سبيل المثال: a + b أو a / b

| OPERATOR | OPERATION |
|---|---|
| + | Addition |
| - | Subtraction |
| * | Multiplication |
| / | Division |
| % | Reminder |

# ASSIGNMENT OPERATORS

يوجد نوعان من assignments في لغة C :

Simple operators : المتغيرات باستخدام الرمز '='على سبيل المثال: a = 8

Compound assignments : خاصة بلغة C وتتكون من رمزين كما هو موضح في الجدول. يمكن كتابة تعبير بطريقة مختلفة أيضًا ، وهذا الرمز يوفر شفرة آلة أكثر كفاءة اي يوفر العبئ على المترجم والذاكرة وسرعة التنفيذ.

| OPERATOR | EXAMPLE | |
|---|---|---|
| | Expression | Equivalent |
| += | a += 8 | a = a + 8 |
| -= | a -= 8 | a = a - 8 |
| *= | a *= 8 | a = a * 8 |
| /= | a /= 8 | a = a / 8 |
| %= | a %= 8 | a = a % 8 |

# INCREMENT AND DECREMENT OPERATORS

يتم الإشارة إلى الزيادة والنقصان Increment and decrement بواسطة 1 للعمليات من خلال "++" و " - -". يمكن ان تسبق او تاتي بعد الرقم كما سنرى

في الحالة الأولى (x++) ، سيتم زيادة المتغير x أولاً ، ثم يتم استخدامه في التعبير.

خلاف ذلك اي (x++) ، سيتم استخدام المتغير لأول مرة في التعبير ، ثم زيادة بمقدار 1. وينطبق الشيء نفسه على عملية الإنقاص

| OPERATOR | EXAMPLE | DESCRIPTION |
|---|---|---|
| ++ | ++a | Variable "a" is incremented by 1 |
| | a++ | |
| -- | --b | Variable "b" is decremented by 1 |
| | b-- | |

# RELATIONAL OPERATORS

تستخدم المعاملات العلائقية في المقارنات لغرض مقارنة متغيرين يمكن أن يكونا أعدادا صحيحة (int) أو أرقام نقطية (float) , إذا تم تقييم التعبير إلى true ، فسيتم إرجاع 1. خلاف ذلك ، يتم إستدعاء القيمة 0. يستخدم هذا في التعبيرات مثل "إذا كان التعبير صحيحًا ..الخ

| OPERATOR | MEANING | EXAMPLE | TRUTH CONDITION |
|---|---|---|---|
| > | is greater than | b > a | if **b** is greater than **a** |

| | | | |
|---|---|---|---|
| >= | is greater than or equal to | a >= 5 | If **a** is greater than or equal to **5** |
| < | is less than | a < b | if **a** Is less than **b** |
| <= | is less than or equal to | a <= b | if **a** Is less than or equal to **b** |
| == | is equal to | a == 6 | if **a** Is equal to **6** |
| != | is not equal to | a != b | if **a** Is not equal to **b** |

## LOGIC OPERATORS

هناك ثلاثة أنواع من العمليات المنطقية في لغة :C المنطق AND والمنطق OR و والنفي .(NOT)

يتم تمثيل الحالات المنطقية في الجدول ادناه

logic zero (0=false)

logic one (1=true)

تشير العمليات المنطقية إلى حقيقة التعبير الكامل. على سبيل المثال: 1 && 0 هو نفسه (تعبير صحيح) && (تعبير زائف) النتيجة هي 0 ، أي خطأ

| **OPERATOR** | **LOGICAL AND** | | |
|---|---|---|---|
| | Operand1 | Operand2 | Result |
| && | 0 | 0 | 0 |
| | 0 | 1 | 0 |

| | | | |
|---|---|---|---|
| | 1 | 0 | 0 |
| | 1 | 1 | 1 |

| **OPERATOR** | **LOGICAL OR** | | |
|---|---|---|---|
| | Operand1 | Operand2 | Result |
| | 0 | 0 | 0 |
| `\|\|` | 0 | 1 | 1 |
| | 1 | 0 | 1 |
| | 1 | 1 | 1 |

| **OPERATOR** | **LOGICAL NOT** | |
|---|---|---|
| | Operand1 | Result |
| `!` | 0 | 1 |
| | 1 | 0 |

# BITWISE OPERATORS

يتم استخدام معاملات Bitwise لتعديل وحدات البت للمتغير. وهي مدرجة في الجدول أدناه:

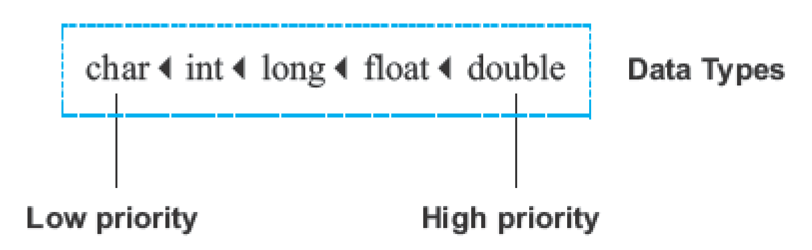
| OPERAND | MEANING | EXAMPLE | RESULT | |
|---|---|---|---|---|
| ~ | Bitwise complement | a = ~b | b = 5 | a = -5 |
| << | Shift left | a = b << 2 | b = 11110011 | a = 11001100 |
| >> | Shift right | a = b >> 2 | b = 11110011 | a = 00011110 |
| & | Bitwise AND | c = a & b | a = 11100011 b = 11001100 | c = 11000000 |
| \| | Bitwise OR | c = a \| b | a = 11100011 b = 11001100 | c = 11101111 |
| ^ | Bitwise EXOR | c = a ^ b | a = 11100011 b = 11001100 | c = 00101111 |

كل مشغل OPERATOR له اولوية كما هو موضح في الجدول.

<table>
<tr><th>PRIORITY</th><th>OPERATORS</th><th>ASSOCIATIVITY</th></tr>
<tr><td>High</td><td><code>() [] -> .</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>! ~ ++ -- +(unary) -(unary) *Pointer &Pointer</code></td><td>from right to left</td></tr>
<tr><td></td><td><code>* / %</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>+ -</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>< ></code></td><td>from left to right</td></tr>
<tr><td></td><td><code>< <= > >=</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>== !=</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>&</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>^</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>|</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>&&</code></td><td>from left to right</td></tr>
<tr><td></td><td><code>||</code></td><td>from right to left</td></tr>
<tr><td></td><td><code>?:</code></td><td>from right to left</td></tr>
<tr><td>Low</td><td><code>= += -= *= /= /= &= ^= |= <= >=</code></td><td>from left to right</td></tr>
</table>

# التحويل بين انواع البيانات DATA TYPE CONVERSION

يتم وضع أنواع البيانات الرئيسية في ترتيب معين على النحو التالي:




إذا تم استخدام معاملين من نوع مختلف في عملية حسابية ، يتم تحويل نوع المعامل ذو الأولوية الأقل lower priority تلقائيًا إلى نوع المعامل ذو الأولوية الأعلى higher priority

في التعبيرات الخالية من التعريف من عملية التعيين ، يتم الحصول على النتيجة بالطريقة التالية:

إذا كان المعامل ذو الأولوية العليا من النوع double ، فإن أنواع جميع المعاملات الأخرى في التعبير بالإضافة إلى النتيجة يتم تحويلها تلقائيًا إلى كتابة double

إذا كان المعامل ذو الأولوية العليا من النوع الطويلlong ، فإن أنواع جميع المعاملات الأخرى في التعبير بالإضافة إلى النتيجة يتم تحويلها تلقائيًا تتحول إلى نوع long

إذا كانت المعاملات طويلة long أو char ، يتم تلقائياً تحويل أنواع كافة المعاملات الأخرى في التعبير لهم , وكذلك النوع int.

```
int x;      // Variable x is declared as integer int
x = 3;      // Variable x is assigned value
x += 3.14; // Number PI (3.14) is added to variable x by performing
            // the assignment operation
```

```
/* The result of addition is 6 instead of expected 6.14. To obtain the
expected result without truncating the numbers following the decimal
point, common addition should be performed (x+3.14), . */
```

## العبارات الشرطية CONDITIONAL OPERATORS

الشرط هو عنصر شائع في اي برنامج. اي من الضروري إجراء واحد من عدة عمليات. وبعبارة أخرى 'إذا تم استيفاء الشرطif (...) ، فقم بـ do (...). وإلا ، إذا لم يتم استيفاء الشرطif-else ، فافعلdo (...) ". يتم استخدام المعاملات الشرطية
if-else و switch في العمليات الشرطية حسب اوامر المستخدم

## CONDITIONAL OPERATOR if-else

يمكن أن تستخدم هذه الجملة الشرطية في شكلين : if وحدها أو if-else. في ما يلي الصيغة العامة باستخدام if :

```
if(expression) operation;
```

- إذا كانت نتيجة التعبير المحصورة بين قوسين ليست 0 (صحيح) ، يتم تنفيذ العملية ويستمر البرنامج في التنفيذ.
- إذا كانت نتيجة التعبير هي 0 (خطأ) ، فلن يتم تنفيذ العملية ويستمر البرنامج على الفور في التنفيذ.

لكن ماذا اذا اردنا ان نعمل خيارات اضافية للبرنامج , مثلا اذا لم تفعل يتنفذ الامر 1 ينفذ الامر 2 , من هنا نوضح الصيغ العامة لجملة if-else

```
if(expression)
operation1
else
operation2
```

بمعنى اخر إذا كانت نتيجة التعبير ليست 0 (صحيح) ، يتم تنفيذ العملية 1 ، وإلا يتم تنفيذ العملية 2. بعد تنفيذ أي من العمليتين ، يستمر البرنامج في التنفيذ

في الاوامر المتعددة ممكن استخدام الصيغة التالية :

```
if(expression) {
... //
... // operation1
...}//
else
operation2
```

يمكن كتابة عامل الشرط if-else باستخدام المشغل الشرطي "؟:" كما في المثال أدناه:

```
(expression1)? expression2 : expression3
```

إذا كان expression1 ليس 0 (true) ، فستكون نتيجة التعبير مساوية للنتيجة التي تم الحصول عليها من expression2.

إذا كانت expression1 هي 0 (false) ، فسوف تكون نتيجة التعبير الكامل مساوية للنتيجة التي تم الحصول عليها منexpression3 .

```
maximum = (a > b)? a : b // Variable maximum is assigned the value of
                         // larger variable (a or b)
```

## Switch OPERATION

على عكس العبارة if-else التي تجعل الاختيار بين خيارين في البرنامج ، يمكّنك من خلال switch من الاختيار بين عدة عمليات متعددة . صيغة switch هي:

```
switch (selector)           // Selector is of char or int type
{
    case constant1:
        operation1          // Group of operators are executed if
        ...                 // selector and constant1 are equal
        break;
    case constant2:
        operation2          // Group of operators are executed if
        ...                 // selector and constant2 are equal
        break;
        ...
    default:
        expected_operation // Group of operators are executed if no
        ...                 // constant is equal to selector
        break;
}
```

يتم تنفيذ عملية التبديل switch بالطريقة التالية: يتم تنفيذ المحدد أولاً ومقارنته بالقيمة constant1

إذا تم العثور على تطابق ، فسيتم تنفيذ العبارات في كتلة هذه الحالة.

إذا لم يتم العثور على تطابق ، فإن المقارنة تتم مقارنة مع constant2 ، وإذا تم العثور على مطابقة ، فسيتم تنفيذ العبارات في هذه الحالة .

إذا لم يتطابق المحدد مع أي ثابت ، فسيتم تنفيذ العمليات التي تقع في default.

من الممكن أيضًا مقارنة تعبير expression مع مجموعة من الثوابتconstants بحيث إذا كان يتطابق مع أي منها ، فسيتم تنفيذ العمليات المناسبة

مثال على ذلك :

```
switch (number) // number represents one day in a week. It is
                // necessary to determine whether it is a week-
{               // day or not.
    case1:case2:case3:case4:case5: LCD_message = 'Weekday'; break;
    case6:case7: LCD_message = 'Weekend'; break;
    default:
    LCD_message_1 = 'Choose one day in a week'; break;
}
```

# الدوران / التكرار PROGRAM LOOP

غالباً ما يكون من الضروري تكرار عملية معينة لبضع مرات في البرنامج. تسمى مجموعة من الأوامر التي يتم تكرارها حلقة البرنامج.وعدد المرات التي سيُنفذ فيها ، أي طول مدة بقاء البرنامج في الحلقة ، يعتمد على شروط الخروج من الحلقة.

## While LOOP

الصيغة العامة لها هي :

```
while(expression){
    commands
    ...
}
```

يتم تنفيذ الأوامر بشكل متكرر (يبقى البرنامج في الحلقة) حتى يصبح التعبير.false

إذا كان التعبير خاطئًا عند الدخول إلى الحلقة ، فلن يتم تنفيذ الحلقة وسيبدأ البرنامج من نهاية الحلقة. نوع خاص من حلقة البرنامج هو الحلقة اللانهائية. يتم تشكيلها إذا بقيت الحالة دون تغيير في الحلقة.

التنفيذ بسيط في هذه الحالة حيث أن النتيجة بين الأقواس تكون دائماً صحيحة (1 = 1) ، مما يعني أن البرنامج يستعيد نفسه في الحلقة نفسها:

```
while(1){
    ... // Expressions enclosed within curly brackets will be
    ... // endlessly executed (endless loop).
}
```

# For LOOP

الصيغة العامة لها هي :

```
for(initial_expression; condition_expression; change_expression) {
    operations
    ...
}
```

- تشبه تنفيذ الحلقة while loop ، باستثناء أنه في هذه الحالة يتم تنفيذ عملية تحديد القيمة الأولية (initial_expression) ضمن الإعلان.
- يعيّن initial_expression المتغير الأولي للحلقة ، والذي يتم مقارنته بالعلامة condition_expression قبل الدخول في الحلقة.
- يتم تنفيذ العمليات داخل الحلقة بشكل متكرر وبعد كل تكرار يتم تغيير قيمة التعبير. يستمر التكرار حتى يصبح condition_expression غير صحيح.

مثال للتوضيح :

```
for(k=1; k<5; k++) // Increase variable k 5 times (from 1 to 5)
             // and  operation repeat expression operation every time
    ...
```

يتم تنفيذ العملية خمس مرات. بعد ذلك ، سيتم التحقق من صحتها من خلال التحقق من أن التعبير k <5 خاطئ (بعد 5 تكرارات k = 5) وسيخرج البرنامج من الحلقة.

# Do-while LOOP

الصيغة العامة لها هي :

```
do
operation
while (check_condition);
```

في هذه الحالة ، يتم تنفيذ العملية مرة واحدة على الأقل بغض النظر عما إذا كان الشرط صحيحًا أو خاطئًا حيث يتم تنفيذ check_condition التعبير في نهاية الحلقة.

إذا كانت النتيجة ليست 0 (true) ، يتم تكرار الإجراء. في المثال التالي ، يبقى البرنامج في حلقة do-while حتى يصل المتغير إلى

1E06 (million iterations)

```
a = 0; // Set initial value

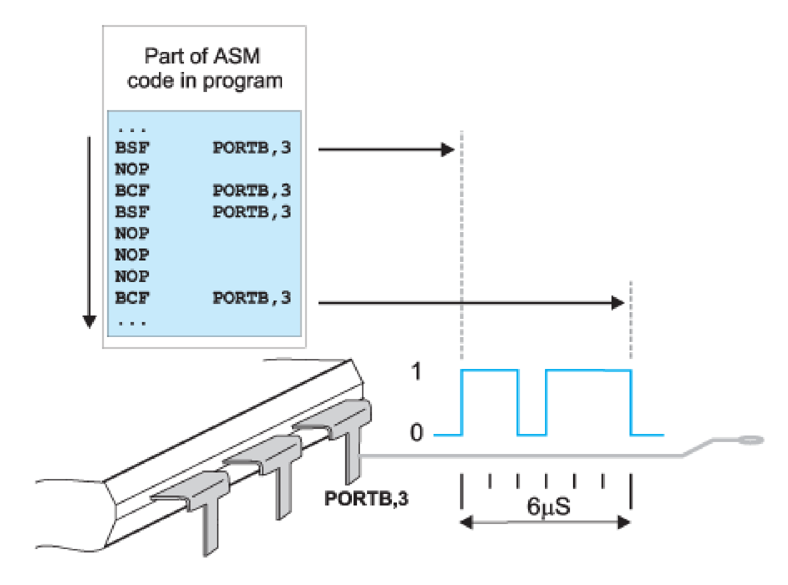
do

a = a+1 // Operation in progress

while (a <= 1E06); // Check condition
```

في بعض الأحيان تتطلب عملية كتابة برنامج في لغة C كتابة أجزاء من الكود في لغة التجميع. يتيح ذلك تنفيذ أجزاء معقدة من البرنامج بطريقة محددة بدقة لفترة زمنية محددة.

على سبيل المثال ، عندما يكون من الضروري وجود نبضات قصيرة جدًا pulses (عدد قليل من الميكروثانية) تظهر دوريًا على دبوس وحدة تحكم دقيقة. في مثل هذه الحالات وما شابه ، فإن أبسط حل هو استخدام رمز التجميع للجزء من البرنامج الذي يتحكم في مدة النبض pulse duration.




يتم إدراج واحد أو أكثر من تعليمات التجميع في البرنامج المكتوب بلغة C باستخدام أمر asm:

```
asm
{
Assembly language instructions
...
}
```

يمكن أن تستخدم الرموز المكتوبة بلغة التجميع الثوابت والمتغيرات المعرفة مسبقًا في لغة .C بالطبع ، بما أن البرنامج بأكمله مكتوب بلغة C ، يتم تطبيق قواعده عند إعلان هذه الثوابت والمتغيرات

```
unsigned char maximum = 100; // Declare variables: maximum = 100
asm
{ // Start of assembly code
    MOVF maximum,W // W = maximum = 100
    ...
} // End of assembly code
```

## المصفوفة ARRAYS

تسمى مجموعة من المتغيرات من نفس النوع مصفوفة , تسمى عناصر المصفوفة بالمكونات components، بينما يسمى نوعها بالنوع الرئيسي. يتم تحديد مصفوفة عن طريق تحديد اسمها name ونوعها type وعدد العناصر number of components التي ستضمها:

الصيغة العامة لها :

```
component_type array_name [number_of_components];
```

يمكن اعتبار المصفوفة كقائمة من المتغيرات من نفس النوع حيث يتم تعيين رقم ترتيبي لكل منها (يبدأ الترقيم دائمًا عند صفر).

غالباً ما تسمى هذه المصفوفة المتجه vector. يوضح الجدول أدناه صفيفًا اسمه shelf يتكون من 100 عنصر

| ARRAY "SHELF" | ELEMENTS OF ARRAY | CONTENTS OF ELEMENT |
|---|---|---|
| 7 | shelf[0] | 7 |
| 23 | shelf[1] | 23 |
| 34 | shelf[2] | 34 |
| 0 | shelf[3] | 0 |
| 0 | shelf[4] | 0 |
| 12 | shelf[5] | 12 |
| 9 | shelf[6] | 9 |

| | | |
|---|---|---|
| ... | ... | ... |
| ... | ... | ... |
| 23 | shelf [99] | 23 |

تمثل محتويات المتغير (عنصر المصفوفة) عددًا من المنتجات التي يحتوي عليها العامود.

يتم الوصول إلى العناصر عن طريق اسلوب العنونة ، أي عن طريق تحديد رقمها الترتيبي

```
shelf[4] = 12;    // 12 items is ‘placed’ on shelf [4]
temp = shelf [1]; // Variable shelf[1] is copied to
                  // variable temp
```

يمكن تخصيص العناصر خلال تعريف الصفيف في المصفوفة في المثال التالي ، يتم تعريف الصفيف المسمى calendar ويتم تعيين كل عنصر عدد معين من الأيام:

```
unsigned char calendar [12] = {31,28,31,30,31,30,31,31,30,31,30,31};
```

# المصفوفات الثنائية TWO-DIMENSIONAL ARRAY

بصرف النظر عن المصفوفات أحادية البعد التي يمكن اعتبارها كقائمة ، هناك أيضًا مصفوفات متعددة الأبعاد في لغة C. في الجمل التالية ، سنصف فقط صفائف ثنائية الأبعاد تسمى المصفوفات ويمكن اعتبارها جداول.

يتم التصريح عن صفيف ثنائي الأبعاد عن طريق تحديد نوع بيانات الصفيف ، واسم الصفيف وحجم كل بُعد. الصيغة العامة :

```
component_type array_name [number_of_rows] [number_of_columns];
```

يمثل number_of_rows و number_of_columns عدد الصفوف والأعمدة في الجدول ، على التوالي.

انظر المثال ادناه , يمثل مصفوفة من نوع int وباسم table

```
int Table [3][4]; // Table is defined to have 3 rows and 4 columns
```

تمثيل هذه المصفوفة في شكل جدول :

| | | | |
|---|---|---|---|
| table[0][0] | table[0][1] | table[0][2] | table[0][3] |
| table[1][0] | table[1][1] | table[1][2] | table[1][3] |
| table[2][0] | table[2][1] | table[2][2] | table[2][3] |

يمكن تخصيص عناصر المصفوفة أثناء تعريف المصفوفة. في المثال كالتالي ، يتم تعيين عناصر تاجدول اي المصفوفة ثنائية الأبعاد. كما رأينا :

```
int Table[2][3] = {{3,42,1}, {7,7,19}};
```

يمكن أيضًا تمثيل المصفوفة أعلاه في شكل جدول تحتوي عناصره على القيم التالية:

| | | |
|---|---|---|
| 3 | 42 | 1 |
| 7 | 7 | 19 |

## الدوال / الاقترنات FUNCTIONS

يتكون كل برنامج مكتوب بلغة C من عدد كبير أو صغير من الوظائف functions. والفكرة الرئيسية هي تقسيم البرنامج إلى عدة أجزاء باستخدام هذه الوظائف من أجل تنفيذ البرنامج بشكل أسهل. إلى جانب ذلك ، تمكننا الوظائف من استخدام مهارات ومعرفة المبرمجين الآخرين. على سبيل المثال ، إذا كان من الضروري إرسال جملة إلى شاشة عرض LCD ، فمن الأسهل استخدام الجزء المكتوب مسبقًا من البرنامج بدلاً من البدء من جديد !.

تتكون الوظائف من الأوامر التي تحدد ما يجب القيام به عند المتغيرات. كقاعدة عامة ، من الأفضل أن يكون لديك برنامج يتكون من عدد كبير من الوظائف البسيطة أكثر من عدد قليل من الوظائف الكبيرة.

يتكون عادةً جسم الدالة من عدة أوامر يتم تنفيذها حسب الترتيب المحدد لها. يجب تعريف كل وظيفة بشكل صحيح حتى يتم تفسيرها بشكل صحيح أثناء عملية التجميع. يحتوي التعريف على العناصر التالية:

- Function name - اسم الدالة
- Function body - هيكل بناء الدالة
- List of parameters - قائمة المغيرات والعوامل
- Declaration of parameters -تعريف المتغيرات والتفاصيل المدرجة
- Type of function result - نوع النتائج الناتجة من الدالة

هكذا تكون شكل الدالة :

```
type_of_result function_name (type argument1, type argument2,...)
{
    Command;
    Command;
    ...
}
```

مثال :

```
/* Function computes the result of division of the numerator num
ber by the denominator
denom. The function returns a structure of type div_t. */


div_t div(int number, int denom);
```

لاحظ أن الدالة لا تحتاج إلى أن تحتوي على معلماتparameters ، ولكن يجب أن تحتوي على أقواس لاستخدامها في إدخالها. خلاف ذلك ، فإن المجمع قد لا يتعرف

عليها كوظيفة functionاي بعد تنفيذها لا ترجع أي نتيجة إلى البرنامج الرئيسي أو إلى الوظيفة التي صمم لاجلها .

يستمر البرنامج في التنفيذ بعد مواجهة قوس مجعد. يتم استخدام هذه الوظائف عندما يكون من الضروري تغيير حالة دبابيس خرج وحدة التحكم الدقيقة ، أثناء نقل البيانات عبر الاتصال التسلسلي serial communication ، عند كتابة البيانات على شاشة LCD إلخ. يتعرف المحول البرمجيcompiler على هذه الوظائف حسب نوع النتيجة المحددة باستخدام Void

```
void function_name (type argument1, type argument2,...)
{
Commands;
}
```

انظر المثال التالي :

```
void interrupt() {
    cnt++ ;          // Interrupt causes cnt to be incremented by 1
    PIR1.TMR1IF = 0; // Reset bit TMR1IF
}
```

يجب أن يحتوي كل برنامج مكتوب بلغة C على دالة واحدة تسمى "main" والتي لا يجب وضعها في بداية البرنامج.

إذا كان من الضروري استدعاء دالة بإرجاع النتائج بعد تنفيذها ، فيتم استخدام أمر الإرجاع return command، الذي يمكن اتباعه بأي تعبير :

```
type_of_result function_name (type argument1, type argument2,...)
{
    Commands;
    ...
    return expression;
}
```

إذا احتوت الدالة على أمر الإرجاع return دون أن يتبعها تعبير ، تتوقف الدالة عن تنفيذها عندما تصادف هذا الأمر ويستمر البرنامج في التنفيذ من الأمر الأول لحد إغلاق القوس مجعد.

## تعريف دالة جديدة DECLARATION OF A NEW FUNCTION

بصرف النظر عن الوظائف التي تعترف بها لغة C تلقائيًا ، هناك أيضًا وظائف جديدة تمامًا يتم استخدامها في البرامج.

يجب الإعلان عن كل وظيفة "غير معرفة " في بداية البرنامج. يسمى تعريف الدالة نموذج أولي وينظر كما يلي:

```
type_of_result function_name (formal parameters)
{
    description of formal parameters
    definition and declaration
    operators
    ...
}
```

نوع من الوظائف التي لا ترجع قيمة void إذا لم يتم تحديد نوع النتيجة بشكل محدد في البرنامج ، فسيتم اعتباره من النوع int (signed integer).

يوضح المثال التالي دالة تقوم بحساب حجم الأسطوانة:

```
const double PI = 3.14159; // Declare constant PI

float volume (float r, float h) // Declare type float for
{                  // formal parameters r and h
   float v;       // Declare type of result v
   v = PI*r*r*h; // Declare function volume
```

```
    return v;
}
```

إذا كان من الضروري إجراء هذا الحساب لاحقًا في البرنامج (يمكن أن يكون حجم الخزان قيد التنفيذ) ،

استدعاء الوظيفة. خلال عملية التجميع ، سيحول المحول compiler محل المعلمات الشكلية الحقيقية كما هو موضح أدناه:

```
float radius=5, height=10, tank; // declare type float for
...                              // real parameters radius,
...                              // height and tank
tank = volume (radius,height);   // calculate the volume of tank
...                              // by calling the volume function
```

## المكتبات FUNCTION LIBRARIES

يتم تخزين أسماء جميع الوظائف المستخدمة في لغة C في الملف المسمى الراس header هذه الوظائف ، حسب الغرض منها ، يتم فرزها في ملفات أصغر تسمى المكتباتlibraries قبل استخدام أي منها في البرنامج ، من الضروري تحديد ملف الرأس header المناسب باستخدام الأمر #include في بداية البرنامج.

إذا واجه ال compiler وظيفة-دالة غير معروفة أثناء تنفيذ البرنامج ، فسيبحث أولاً عن التعريف الخاص به في المكتبات التي تم تحديدها.

# المكتبات المتوفرة STANDARD ANSI C LIBRARIES

لم تكن وظائف لغة C موحدة في البداية وقام صانعو البرامج بتعديلها وفقًا لاحتياجاتهم. لكن لغة سي أصبحت شائعة جدا في وقت قريب ، وكان من الصعب إبقاء كل شيء تحت السيطرة. كان من الضروري تقديم نوع من المعايير لوضع الأمور في نصابها الصحيح. يسمى المعيار المستخدم ANSI C ويحتوي على 24 مكتبة مع وظائف متعددة لكل مكتبة. عادة ما يتم توفير هذه المكتبات مع كل مترجم C حيث يتم تنفيذ العمليات الأكثر شيوعا باستخدامها.

المكتبات هي :

```
<assert.h>      <complex.h>   <ctype.h>
<errno.h>       <fenv.h>      <float.h>
<inttypes.h>    <iso646.h>    <limits.h>
<locale.h>      <math.h>      <setjmp.h>
<signal.h>      <stdarg.h>    <stdbool.h>
<stdint.h>      <stddef.h>    <stdio.h>
<stdlib.h>      <string.h>    <tgmath.h>
<time.h>        <wchar.h>     <wctype.h>
```

كل ما قرأته حتى الآن حول البرمجة بلغة C هو مجرد نظريات من المفيد أن تعرفه خلال برمجة ال PIC لكن لا تقلق اذا لم تفهم اي نقطة من النقاط اعلاه في التنفيذ ستلاحظ ان الامر ممتع وسهل

ربما سوف تواجه بعض من المشاكل مع أسماء دقيقة للسجلات registers وعناوينهم addresses وأسماء وحدات التحكم الخاصة control bits والكثير غيرها أثناء كتابة البرنامج الأول في لغة C.

لا يكفي أن تكون على دراية بنظرية اللغة C لجعل الميكروكونترولر يقوم بشيء مفيد لا بد من التوسع قليلا في بعض المواضيع وقراءة الكثير من الكتب ! , هذا الكتاب ليس الا البداية .

ذكرنا سابقا أن الميزة الرئيسية للغات البرمجة ذات المستوى العالي مثل C هي أنها تمكنك من استخدام المعرفة والعمل للآخرين.

مكتبات الوظائف هي أفضل مثال على ذلك. إذا كنت بحاجة إلى وظيفة لأداء مهمة معينة أثناء كتابة أحد البرامج ، فكل ما عليك فعله هو البحث عنه في بعض المكتبات المدمجة في المترجم واستخدامه. على سبيل المثال إذا كنت تحتاج إلى دالة لتوليد صوت على بعض الاطراف ، فافتح مكتبة الصوت Sound library في نافذة مدير المكتبة Library Manager وانقر نقرًا مزدوجًا فوق الوظيفة المناسبة Sound_Play. يظهر وصف مفصل لهذه الوظيفة على الشاشة. قم بنسخه إلى برنامجك وقم بتعيين المعلمات المناسبة parameters. في حالة تحديد هذه المكتبة ، سيتم التعرف على وظائفها تلقائيًا أثناء عملية التحويل البرمجي بحيث لا يكون من الضروري استخدام الأمر #include.

تتضمن مكتبات ANSI C الاساسية ما يلي :

| LIBRARY | DESCRIPTION |
| --- | --- |
| ANSI C Ctype Library | Mainly used for testing or data conversion |
| ANSI C Math Library | Used for floating point mathematical operations |
| ANSI C Stdlib Library | Contains standard library functions |
| ANSI C String Library | Used to perform string and memory manipulation operations |

## مكتبات متفرقة

تحتوي المكتبات المتفرقة على بعض وظائف الأغراض العامة غير المدرجة في مكتبات ANSI C الاساسية في الجدول السابق :

| LIBRARY | DESCRIPTION |
|---|---|
| Button Library | Used for a project development |
| Conversion Library | Used for data type conversion |
| Sprint Library | Used for easy data formatting |
| PrintOut Library | Used for easy data formatting and printing |
| Time Library | Used for time calculations (UNIX time format) |
| Trigonometry Library | Used for fundamental trigonometry functions implementation |
| Setjmp Library | Used for program jumping |

## مكتبات دعم العتاد HARDWARE SPECIFIC LIBRARIES

تتضمن المكتبات الخاصة بالأجهزة وظائف تهدف إلى استخدامها للتحكم في تشغيل وحدات hadware المختلفة

| LIBRARY | DESCRIPTION |
|---|---|
| ADC Library | Used for A/D converter operation |
| CAN Library | Used for operation with CAN module |

| | |
|---|---|
| CANSPI Library | Used for operation with external CAN module (MCP2515 or MCP2510) |
| Compact Flash Library | Used for operation with *Compact Flash memory cards* |
| EEPROM Library | Used for operation with built-in EEPROM memory |
| EthernetPIC18FxxJ60 Library | Used for operation with built-in Ethernet module |
| Flash Memory Library | Used for operation with built-in Flash memory |
| Graphic Lcd Library | Used for operation with graphic LCD module with 128x64 resolution |
| I2C Library | Used for operation with built-in serial communication module I2C |
| Keypad Library | Used for operation with keyboard (4x4 push buttons) |
| Lcd Library | Used for operation with LCD display (2x16 characters) |
| Manchester Code Library | Used for communication using *Manchester code* |
| Multi Media Card Library | Used for operation with multimedia MMC flash cards |
| One Wire Library | Used for operation with circuits using *One Wire* serial communication |
| Port Expander Library | Used for operation with port expander MCP23S17 |

| | |
|---|---|
| PS/2 Library | Used for operation with standard keyboard PS/2 |
| PWM Library | Used for operation with built-in PWM module |
| RS-485 Library | Used for operation with modules using RS485 serial communication |
| Software I2C Library | Used for I2C software simulation |
| Software SPI Library | Used for SPI software simulation |
| Software UART Library | Used for UART software simulation |
| Sound Library | Used for audio signal generation |
| SPI Library | Used for operation with built-in SPI module |
| SPI Ethernet Library | Used for SPI communication with ETHERNET module (ENC28J60) |
| SPI Graphic Lcd Library | Used for 4-bit SPI communication with graphic LCD display |
| SPI Lcd Library | Used for 4-bit SPI communication with LCD display (2x16 characters) |
| SPI Lcd8 Library | Used for 8-bit SPI communication with LCD display |
| SPI 6963C Graphic Lcd Library | Used for SPI communication with graphic LCD display |
| UART Library | Used for operation with built-in UART module |
| USB Hid Library | Used for operation with built-in USB module |

حسنًا سوف نكتفي في هذا المقدار في النسخة التجريبية لهذه السلسلة من شرح اساسيات برمجة الميكروكونترولر وسوف نبدأ في تنفيذ مشروع كامل باستخدام الميكروكونترولر ,وذلك لفهم طريقة العمل :

سوف نقوم في بناء دائرة الكترونية للتحكم في وميض مصباح LED ونظرا لوجود أنواع مختلفة من ميكروكنترولر ، اخترنا واحدة لغرضنا. لقد اخترنا ATmega32U2 لأنه يمكننا برمجة ذلك عبر USB ومن السهل بدرجة معقولة اللحام في المنزل.

سنقوم بتصميم دوائرنا من الصفر. لذلك نحن بحاجة إلى تصميم مخطط تخطيطي مع جميع المكونات والتوصيلات اللازمة لجعل دائرتنا تعمل.

ماذا نحتاج؟

كيف يمكننا أن نقرر أي مكونات للاتصال متحكم لدينا؟

دعونا نفكر في ذلك. نحتاج إلى طاقة للشريحة ، وإلا فلن تعمل. ونحن بحاجة إلى اتصال USB ، لأننا سنبرمجه عبر USB. ونحن بحاجة إلى بعض المسامير المادية حيث يمكننا بسهولة توصيل الأشياء بدائرتنا واختبارها.

لذلك نحن بحاجة الى :

- Power
- USB Connection
- Pin Connections

ورقة البيانات

ATmega32U2 لها ورقة بيانات خاصة لمعرفة بالضبط المكونات التي نحتاجها ، وكيفية توصيلها ، ونحن ننظر في مكوناتها من :

http://ww1.microchip.com/downloads/en/DeviceDoc/doc7799.pdf

ورقة البيانات عبارة عن مستند شامل يحتوي على الكثير من البيانات التقنية حول كيفية عمل المتحكم الدقيق وكيف يمكنك التحكم في أجزاء مختلفة منه.

إذا لم تكن معتادًا على قراءة ورقة البيانات ، فقد تشعر أنك غامرة بعض الشيء. ولكن بعد النظر إلى بعض أوراق البيانات ، سوف تبدأ عاجلاً أم آجلاً في فهم كيفية وضعها.

ليس من الضروري قراءة ورقة البيانات من البداية إلى النهاية. تحتاج فقط إلى قراءة الأجزاء المهمة لما تريد صنعه

لذلك إذا كنت ترغب في عمل مؤقت مع متحكمك ، فأنت تقرأ قسم المؤقت. إذا كنت تريد استخدام UART ، فإنك تبحث عن قسم.UART

لاحظ في ورقة البيانات من ATmega32U2 ، يتم وضع جدول المحتويات في النهاية.

- Power and USB

يمكننا اختيار ما إذا كنا نريد تشغيل الدائرة بواسطة كابل USB أو بواسطة كابل طاقة إضافي.

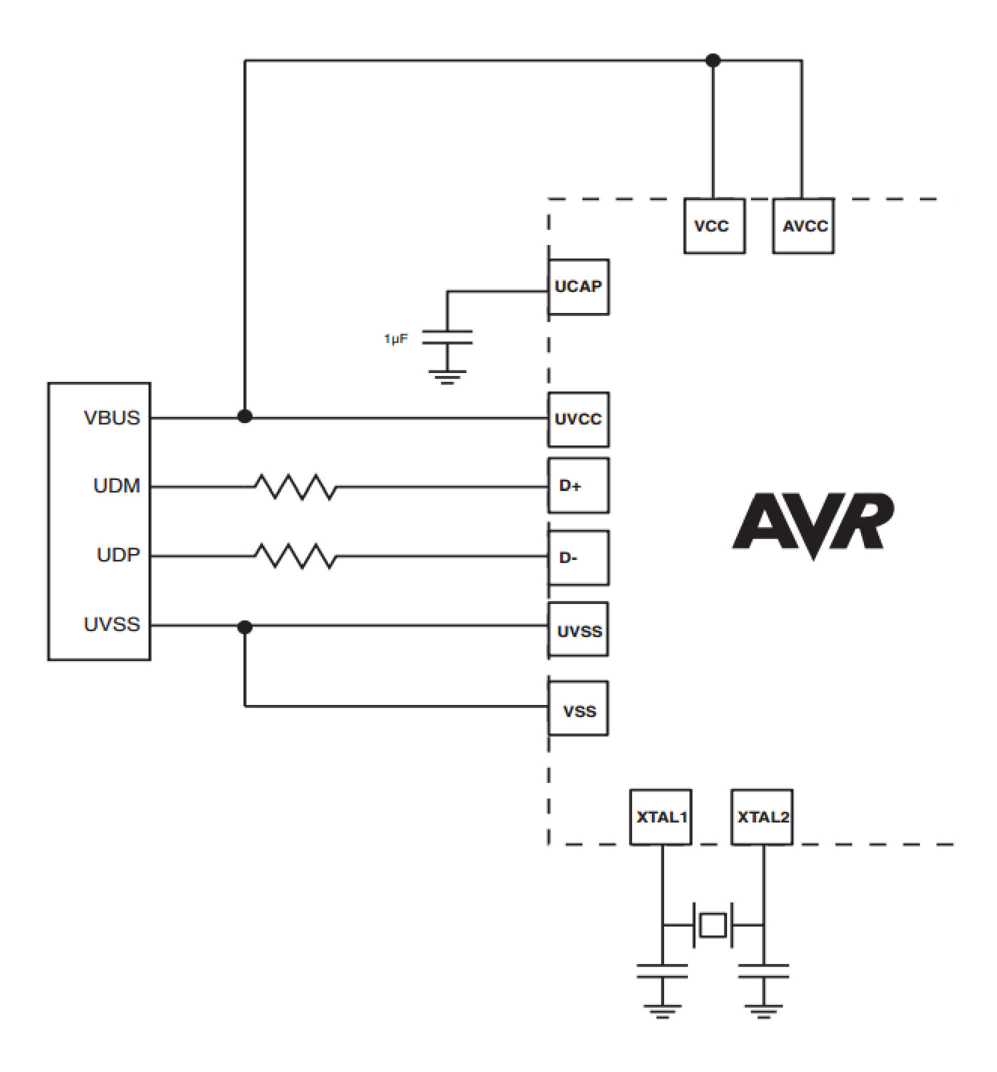
للحفاظ على الأمور بسيطة ومع وجود مكونات أقل ، لنجعل هذا جهازًا يعمل بنظام .USBهذا يعني أنه لن يعمل إلا عند توصيله بـ.USB

فماذا يعني هذا؟ ما المكونات التي نحتاجها؟

لمعرفة ذلك ، دعونا ننظر في قسم USB في ورقة البيانات ، ونرى ما نجده:

خيارات وحدة الطاقة USB

من جدول المحتويات في نهاية ورقة البيانات ، نجد قسمًا باسم خيارات تشغيل وحدة USB في الصفحة 186. هنا يمكننا العثور على الشكل التالي الذي يوضح لنا كيفية توصيل قسم USB بالرقاقة لجعله مدعومًا بواسطة USB .




How to connect the ATmega32U2 as a USB powered 5V device. (Source: Atmel.com)

من إرشادات التصميم في الصفحة 189 ، نتعلم أن المقاومات يجب أن تكون مقاومة 22 اوم ، وأنه يوصى بشدة أن يكون هناك مكثف 10 µF على خط VBUS. لذلك دعونا نضيف مكثف µF10 أيضًا.

في الصورة أعلاه ، يمكننا أن نرى أن هناك أيضًا بلورة واثنين من المكثفات متصلة بدبابيس XTAL1 و .XTAL2 لماذا هذا؟

يحتاج المتحكم إلى ساعة للعمل. تحتوي معظم ميكروكنترولر على مذبذب RC داخلي يخلق إشارة ساعة. وكذلك تفعل.ATmega32U2

ولكن الشيء هو أن جزء USB من متحكم غير قادر على تشغيل من تلك الساعة الداخلية. حتى نجعل USB يعمل ، نحن بحاجة إلى بلورة خارجية.

بالضبط ما هو نوع البلورة الذي نحتاجه ، يمكننا أن نجد في الصفحة 30 في ورقة البيانات. هنا ، يتم سرد القيم مكثف أيضا.

## توصيلات LED

نريد أن نكون قادرين على توصيل الأشياء بدائرة متحكمنا. لذلك ، سنضيف 16 دبوسًا فعليًا متصلين بدبابيس I / O PB0-PB7 و PD0-PD7. هذا يجعل من السهل توصيل شيء ما بدائرتنا.

من الشائع جداً أن يكون لديك زر إعادة الضبط على دائرة متحكم دقيق. هذا يجعل من السهل إعادة ضبط متحكم دون الحاجة إلى فصل كبل.USB

من صفحة 47 من ورقة البيانات ، يمكننا قراءة:

»تتم إعادة ضبط وحدة MCU عند وجود مستوى منخفض على دبوس RESET لفترة أطول من الحد الأدنى لطول النبض«.

لذلك نحن بحاجة إلى التأكد من سحب دبوس إعادة الضبط عادةً عالياً ، وسحبه فقط عند الضغط على زر إعادة الضبط.

للقيام بذلك ، نستخدم شيئًا يسمى المقاوم القابل للسحب.

هذا هو المقاوم متصلة بين دبوس إعادة تعيين و VCC 5 فولت - الذي "يسحب" دبوس إعادة تعيين عالية.

نقوم بتوصيل زر إعادة الضبط بطريقة بحيث يتم توصيل دبوس إعادة الضبط بالأرض (0 فولت) إذا تم الضغط على الزر.

ما القيمة التي يجب أن يكون لدى هذا المقاوم؟ قيمة جيدة حوالي 10 كيلو أوم

يحتوي موقع Sparkfun على مقال جيد حول كيفية اختيار قيمة للمقاومة المنسدلة:

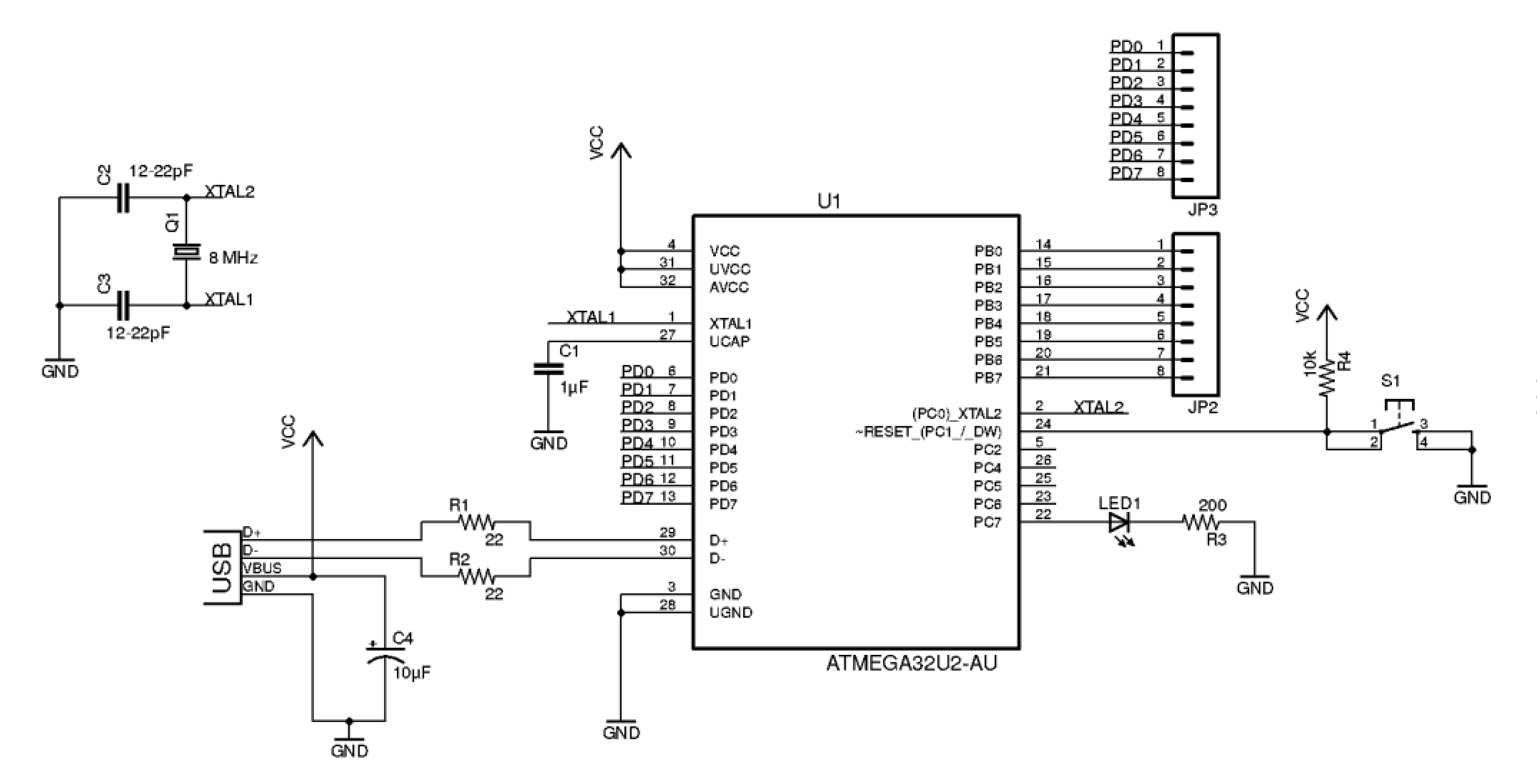
https://learn.sparkfun.com/tutorials/pull-up-resistors

مخطط الدائرة للمتحكم كاملة

إذا قمنا بدمج جميع الأجزاء التي ناقشناها أعلاه ، فسينتهي بنا الأمر إلى مخطط الدوائر التالي. ستلاحظ أن بعض الأجزاء تبدو غير متصلة بالباقي. ولكن ، عندما يكون هناك اسم على السلك ، فهذا يعني أن هذا السلك متصل بجميع الأسلاك الأخرى التي تحمل الاسم نفسه.

وبعد المخطط ماذا سنفعل ؟ سنقوم في تصميم لوحة الدوائر والحصول عليها وهذا سوف ندرسه لاحقًا في هذه السلسلة بعنوان Board Design

كل ذلك يجمع لاحقًا يدويًا . كل ما علينا فعله الآن هو إنشاء لوحة دوائر الكترونية , وفي الصفحة التالية تمثل المخطط للدائرة :

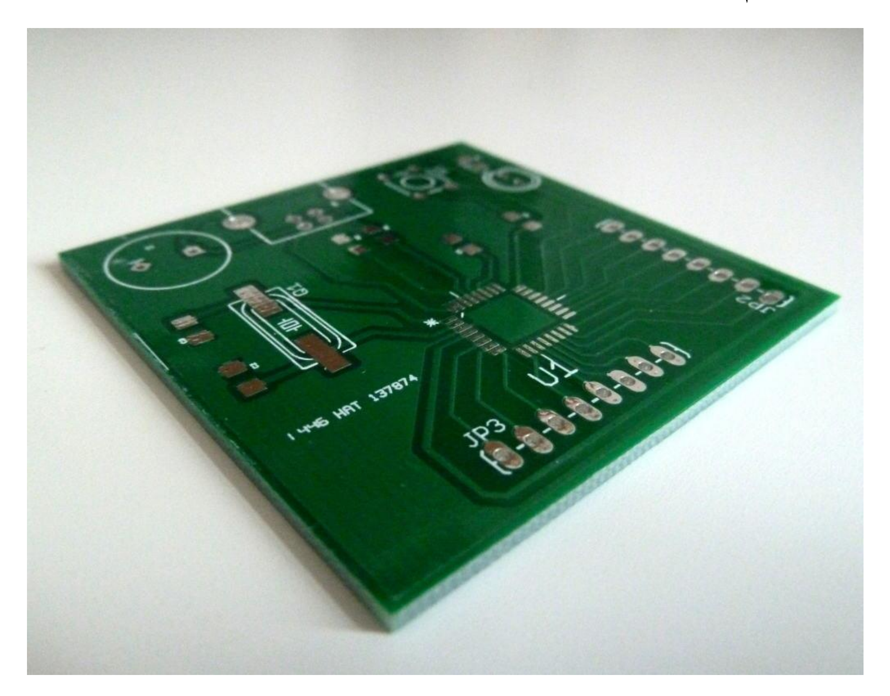


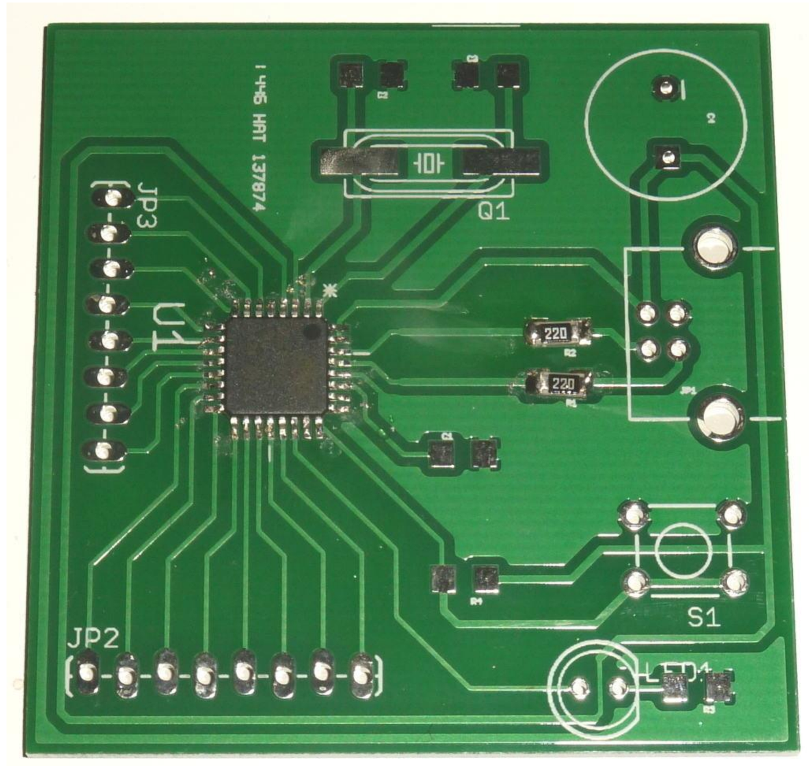
C2
12-22pF
XTAL2
Q1
8 MHz
C3
XTAL1
12-22pF
GND
VCC
U1
VCC
UVCC
AVCC
XTAL1
UCAP
C1
1µF
PD0
PD1
PD2
PD3
PD4
PD5
PD6
PD7
D+
D-
GND
UGND
ATMEGA32U2-AU
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
(PC0)_XTAL2
~RESET_(PC1_/_DW)
PC2
PC4
PC5
PC6
PC7
JP3
JP2
10k
R4
S1
LED1
200
R3
USB
D+
D-
VBUS
GND
R1
22
R2
22
C4
10µF

نحن الآن في المراحل الاخيرة من البرنامج التعليمي للمتحكم القابل للبرمجة. لوحة بدون مكونات ليست مثيرة للاهتمام , لذلك نحن بحاجة إلى الحصول على بعض المكونات أيضًا وهي مدرجة جميعها في الجدول التالي :

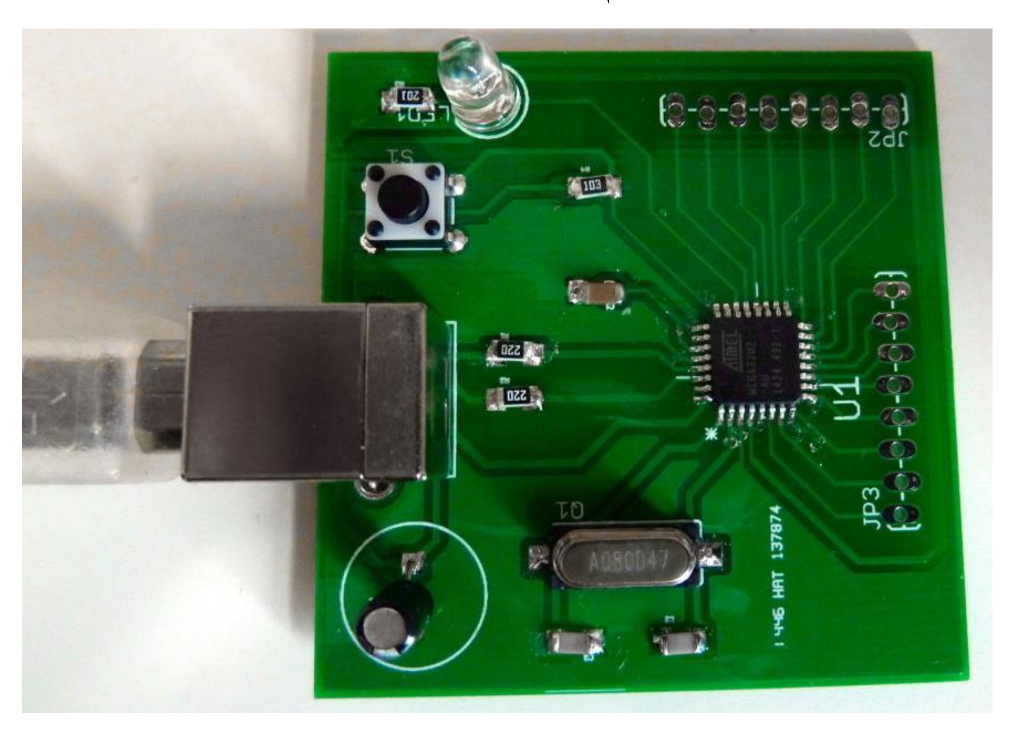
| Part | Description | Value | Package |
|---|---|---|---|
| C1 | Capacitor | 1µF | SMD 1206 |
| C2, C3 | Capacitor | 12-22pF | SMD 1206 |
| C4 | Capacitor Polarized | 10µF | Through-hole |
| JP1 | USB Connector | USB Type B Receptacle | Through-hole |
| JP2, JP3 | Header 8 pin | | Through-hole |
| LED1 | Light Emitting Diode | 1.8V | Through-hole |
| Q1 | Crystal | 8 MHz | SMD C49UP |
| R1, R2 | Resistor | 22 Ohm | SMD 1206 |
| R3 | Resistor | 200 Ohm | SMD 1206 |
| R4 | Resistor | 10k Ohm | SMD 1206 |
| S1 | Momentary Switch | | Through-hole |
| U1 | Microcontroller | ATmega32U2 | TQFP-32 |

سيكون شكل اللوحة التي قمنا بتصميمها كما يلي او يمكنك استخدام اي وسيلة للربط وها نحن بصدد لحام اللوحة وبرمجتها.



## لحام المكونات الأخرى




اختبار الدائرة :

تأتي شريحة ATmega32U2 مع محمل إقلاع مبرمج مسبقًا يجب أن يظهر كجهاز USBعند توصيله بجهاز كمبيوتر.

بعد أن تم لحام كل شيء ، نقوم بفحص مفاصل اللحام عن قرب باستخدام مجهر أو أي شيء آخر , لأنه إذا كانت هناك دوائر قصيرة تسببها نقطة تلحيم صغيرة جدًا في مكان ما ، فقد نتسبب في تلف دائرتنا.

عملية البرمجة :

الآن بعد أن عرفت أن جزء ال USB يعمل ، فقد حان الوقت لبرمجة الدائرة مع بعض الكودات بلغة ميكرو سي .

يمكنك اختيار البرنامج الذي تراه مناسبا وعن طريق الخطوات الثلاثة :

1 – كتابة البرنامج

2 – ترجمة البرنامج الى للغة الشريحة

3 – تحميل البرنامج الى الشريحة

لإجراء اختبار بسيط ، قمت بإنشاء كود وميض LED لا يفعل أكثر من وميض مؤشر LEDعلى اللوحة .

هنا هو البرنامج الذي استخدمته :

```
#define F_CPU 1000000
#include <avr/io.h>
#include <util/delay.h>

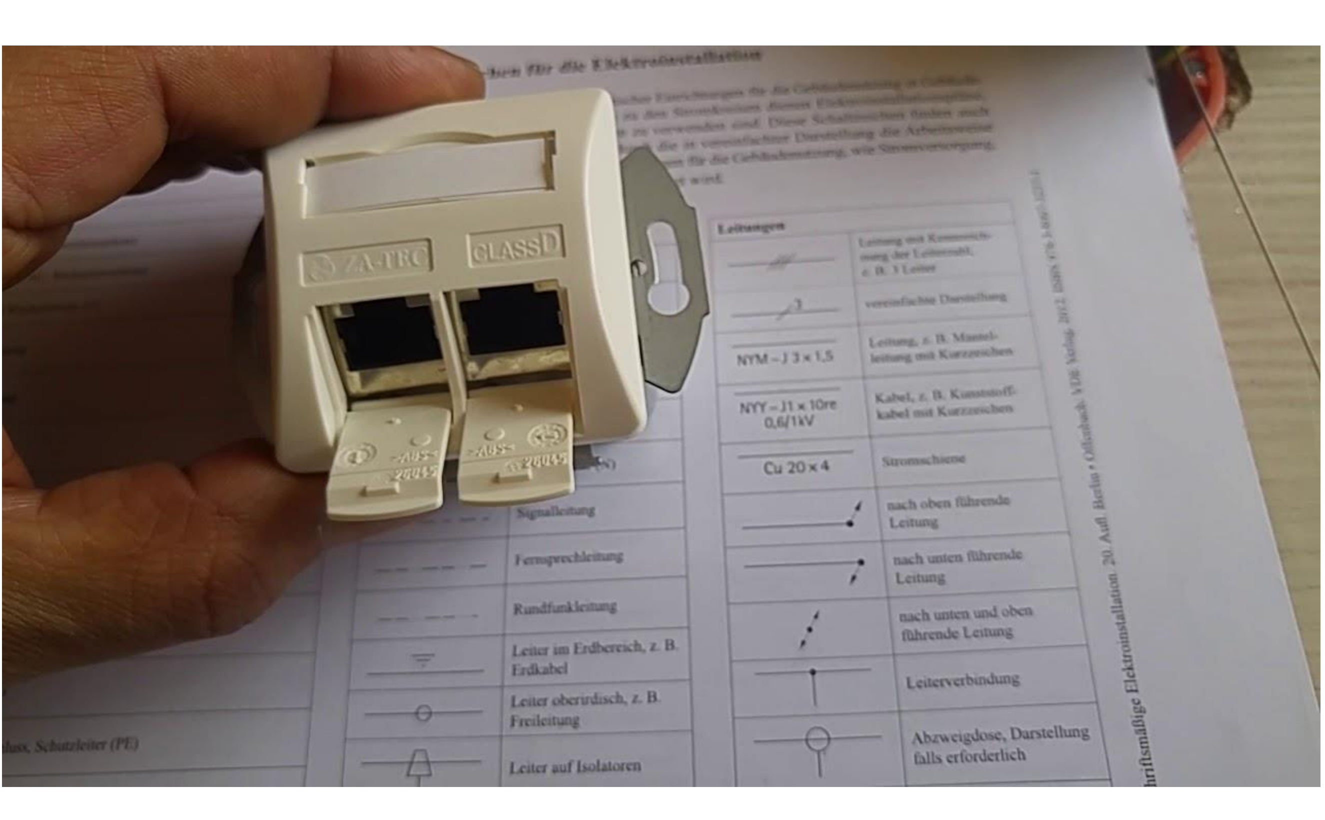
int main(void)
{
    DDRC = (1<<PC7); //Sets the direction of the PC7 to output
    PORTC = (1<<PC7);              //Sets PC7 high

    while(1)
    {
        _delay_ms(500);            //Wait 500 milliseconds
        PORTC &= ~(1<<PC7);        //Turn LED off

       _delay_ms(500);             //Wait 500 milliseconds
       PORTC |= (1<<PC7);          //Turn LED on
    }

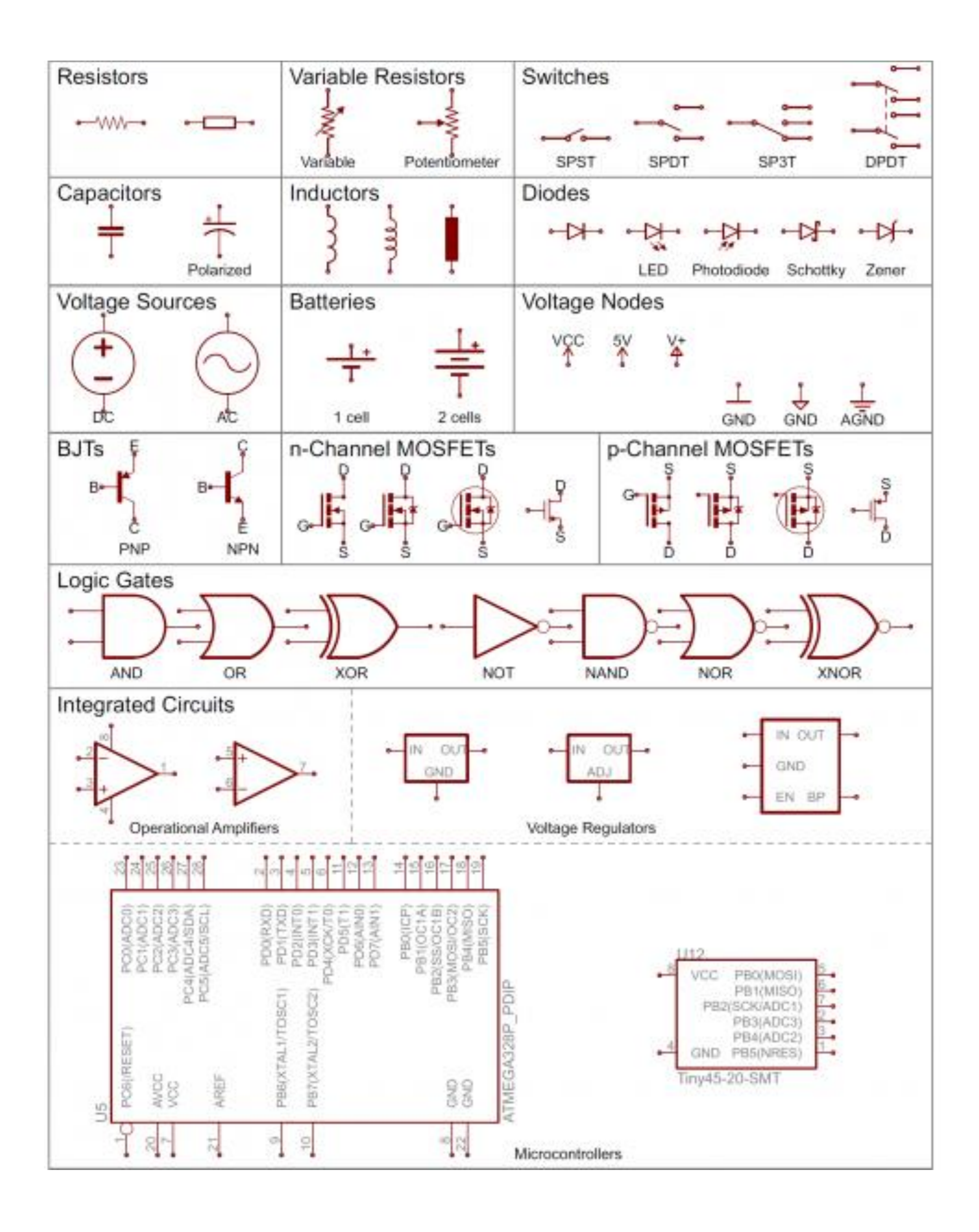
    return 0;
}
```

وعند تحميل الكود بواسطة ال USB ستلاحظ وميض المصباح حسب المدة الزمنية .




تُعد المُخططات هي خريطتنا الخاصة لتصميم أو بناء أو إصلاح الدوائر الكهربية. ولذلك فإن معرفة وفهم كيفية قراءة واتباع المُخططات من أهم المهارات التي يجب أن يتقنها أي مُهتم بالتعامل مع الإلكترونيات.

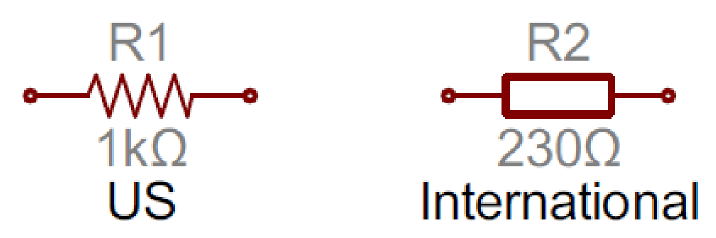
هذا الدرس من شأنه أن يجعل منك شخصاً مُلماً بكيفية قراءة الرسوم التخطيطية، وسنقوم بالمرور على جميع الرموز الرئيسية المُستخدمة في عمل المُخططات.


Resistors
Variable Resistors
Variable
Potentiometer
Switches
SPST
SPDT
SP3T
DPDT
Capacitors
Polarized
Inductors
Diodes
LED
Photodiode
Schottky
Zener
Voltage Sources
DC
AC
Batteries
1 cell
2 cells
Voltage Nodes
VCC
5V
V+
GND
GND
AGND
BJTs
PNP
NPN
n-Channel MOSFETs
p-Channel MOSFETs
Logic Gates
AND
OR
XOR
NOT
NAND
NOR
XNOR
Integrated Circuits
Operational Amplifiers
Voltage Regulators
Microcontrollers
ATMEGA328P_PDIP
Tiny45-20-SMT

هل أنت مستعد لتلقي وابل من رموز مُختلف المكونات المستخدمة في الدوائر الكهربية؟ إليك بعض من الرموز التخطيطية الأساسية القياسية للعديد من تلك المكونات.

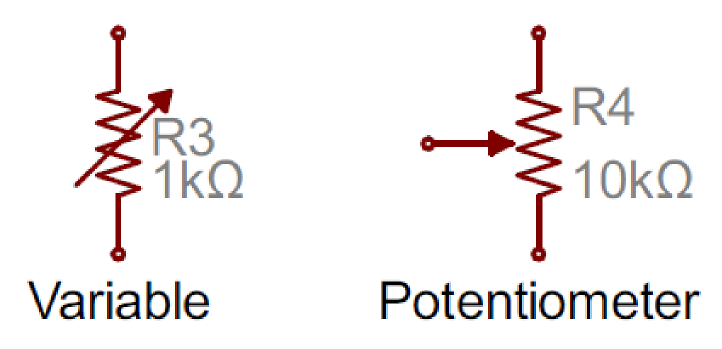
## المُقاومات (Resistors)

المُقاومات من أكثر المكونات الأساسية استخداماً في الدوائر الكهربية. وفي الغالب يُرمز للمقاومات في المُخططات بخط متعرج له طرفين يخرجان منه. من الممكن في المُخططات التي تستخدم الرموز العالمية (international symbols) أن يتم استخدام مُستطيل كرمز للمقاومات بدلاً من الخط المتعرج.



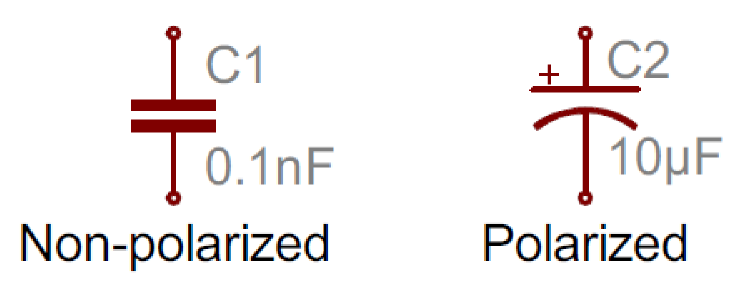

## مقاييس الجهد الانزلاقية (Potentiometers) والمقاومات المتغيرة (Variable Resistors)

في مقاييس الجهد الانزلاقية والمُقاومات المتغيرة يتم إضافة سهم للرمز الخاص بالمُقاومات العادية (الخط المتعرج). المُقاومات المتغيرة هي مُكونات ذات طرفين فقط لذلك يتم وضع السهم قطرياً عبر منتصف الخط المتعرج. أم مقاييس الجهد الانزلاقية فهي مكونات ذات ثلاثة أطراف، لذلك يُشكل السهم الطرف الثالث (المنزلق (wiper)).



## المُكثفات (Capacitors)

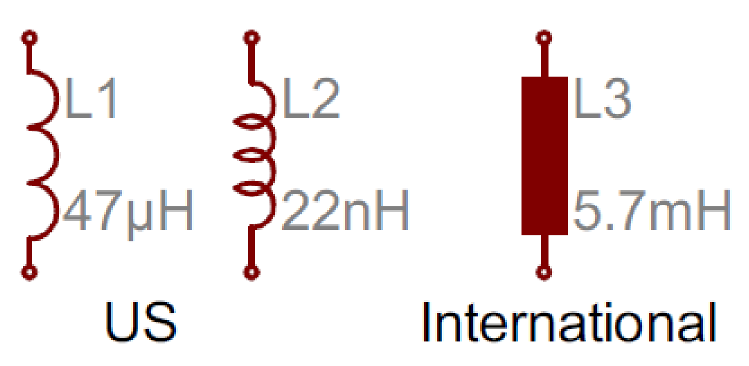
هناك رمزان شائعا الاستخدام للمكثفات. الرمز الأول يُمثل المكثفات المُستقطبة (polarized)والتي تكون في العادة مُكثفات إلكتروليتية (electrolytic) أو مصنوعة من التنتاليوم(tantalum) ، والرمز الآخر يُستخدم مع المُكثفات الغير مُستقطبة .(non-polarized) يوجد طرفان في كلا الرمزين متعامدان على لوحين




الرمز الذي يحتوي على لوح مقوس يُستخدم للدلالة على المُكثفات المُستقطبة. اللوح المقوس يُمثل مهبط (cathode) المُكثف (الطرف السالب)، والذي يكون له جهد منخفض عن الطرف الموجب للمكثف (المصعد (anode)). من الممكن أيضاً أن يتم إضافة علامة زائد (+) للطرف الموجب من المُكثف المُستقطب.

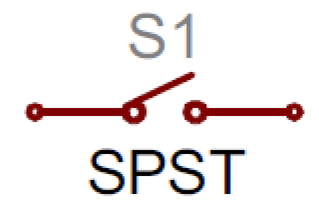
## ملفات الحث (Inductors)

في الغالب يتم تمثيل ملفات الحث إما من خلال خط به عدة مطبات منحنية أو من خلال خط على شكل لفات لولبية. أما في الرموز العالمية فيتم الإشارة إلى ملفات الحث باستخدام مستطيل مُظلل.

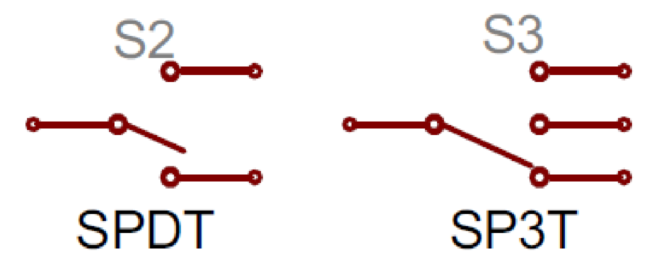


## المفاتيح (Switches)

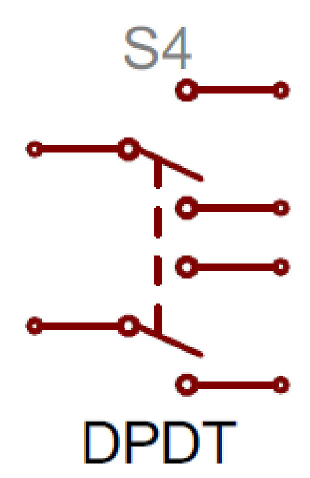
توجد المفاتيح في العديد من الأشكال. أبسط أنواع المفاتيح هو المفتاح الذي يحتوي على نقطة تلامس واحدة وتحويلة واحدة (single-pole/single-throw (SPST))، وهو يحتوي على طرفين مع خط متصل بأحدهما ومنفصل عن الآخر يُمثل المُشغل الميكانيكي (actuator) (الجزء الذي يوصل الطرفين معاً).




المفاتيح التي تحتوي على تحويلة واحدة وأكثر من نقطة تلامس، مثل مفاتيح SPDT وSP3T تحتوي على المزيد من الأطراف التي يُمكن أن يتصل بها المُشغل الميكانيكي.




أما المفاتيح التي تحتوي على عدة تحويلات فغالباً ما تتكون من عدة مفاتيح متشابهة مع وجود خط متقطع يصل بين منتصف المُشغلات الميكانيكية.

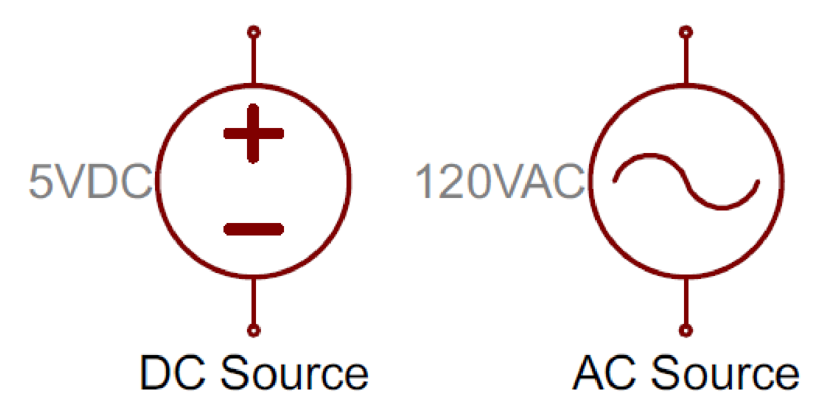


## مصادر الطاقة (Power Sources)

هناك الكثير من الخيارات مُتاحة لتوصيل الطاقة لأي مشروع، وبالمثل هناك العديد من الرموز المُستخدمة لتمثيل المصادر المُختلفة للطاقة في مُخططات الدوائر الكهربية.

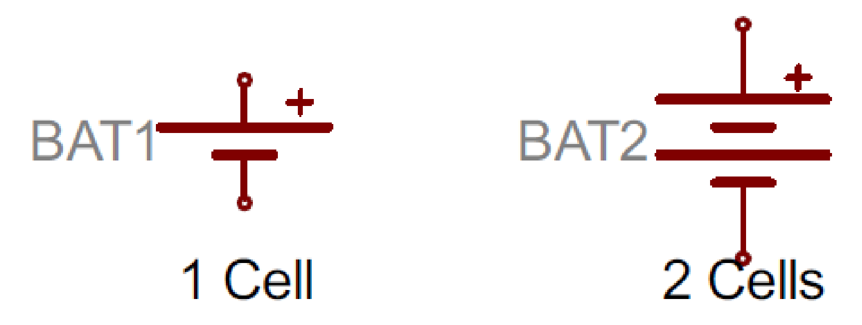
## مصادر الجهد المُستمر (DC Voltage) والجهد المتردد (AC Voltage)

في معظم الأوقات التي تتعامل فيها مع الإلكترونيات تقوم باستخدام مصادر جهد ثابت. من الممكن استخدام أي من الرمزين التاليين لتحديد ما إذا كان المصدر يعطي تيار متردد (Alternating current (AC)) أم تيار مُستمر(direct current (DC))




## البطاريات (Batteries)

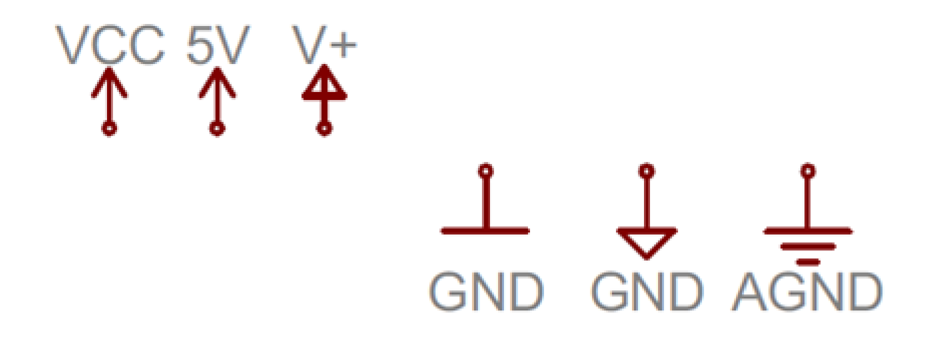
سواء كانت البطاريات من النوع القلوي (Alkaline AA) الذي يكون على شكل اسطوانة، أو من نوع الليثيوم بوليمر (lithium-polymer) الذي يكون على شكل مستطيل، يتم الرمز إليها باستخدام زوج من الخطوط المتوازية الغير متساوية في الطول.



عند توصيل عدة بطاريات معاً على التوالي يتم استخدام عدد أزواج الخطوط يساوي عدد البطاريات. كما أنه في الغالب يتم استخدام الخط الطويل لتمثيل الطرف الموجب، والخط القصير لتمثيل الطرف السالب.

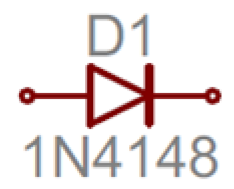
## عُقَد الجهد (Voltage Nodes)

في بعض الأحيان –خاصة في المُخططات التي تكون مُزدحمة بالرموز- من الممكن أن تقوم بتخصيص رموز خاصة لقيم الجهد عند العُقَد .(node voltages) من الممكن أن تقوم بتوصيل المكونات بهذه الرموز ذات الطرف الواحد لكي تكون مُتصلة مباشرة بجهد 5 Vأو 3.3 Vأو VCC أو بالأرضي .(GND) عقد الجهد الموجبة عادة ما تحتوي على سهم يُشير لأعلى، بينما عقد الأرضي عادة ما تحتوي على واحد إلى ثلاثة خطوط (وأحياناً تحتوي على سهم يُشير لأسفل أو مثلث)




## الديودات (Diodes)

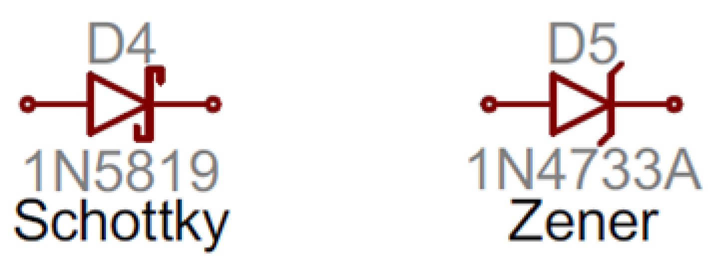
في العادة يتم تمثيل الديودات العادية بواسطة مثلث يحتوي على خط عند رأسه موازي لقاعدته. الديودات مكونات مُستقطبة، لذلك لا بد من وجود طريقة للتمييز بين طرفيها. الطرف الموجب (المصعد) هو الطرف جهة قاعدة المثلث. أما الطرف السالب (المهبط) فهو الطرف جهة رأس المثلث (جهة الخط).



هناك أنواع كثيرة ومتنوعة من الديودات، رمز كل منها يحتوي على إضافة خاصة على الرمز الخاص بالديود العادي. في الديودات المُضيئة (LEDs) يتم إضافة سهمين يشيران للخارج على رمز الديود العادي. وفي الديودات الضوئية (Photodiodes) التي تقوم بتوليد الطاقة من الضوء (هي في الأساس خلايا شمسية صغيرة) يتم إضافة سهمين يُشيران للداخل على رمز الديود العادي.




الأنواع الأخرى من الديودات مثل ديود شوتكي (Schottky) أو ديود زينر (Znener)لها رموز خاصة بها تحتوي على تعديلات في الخط الموجود في رمز الديود العادي.

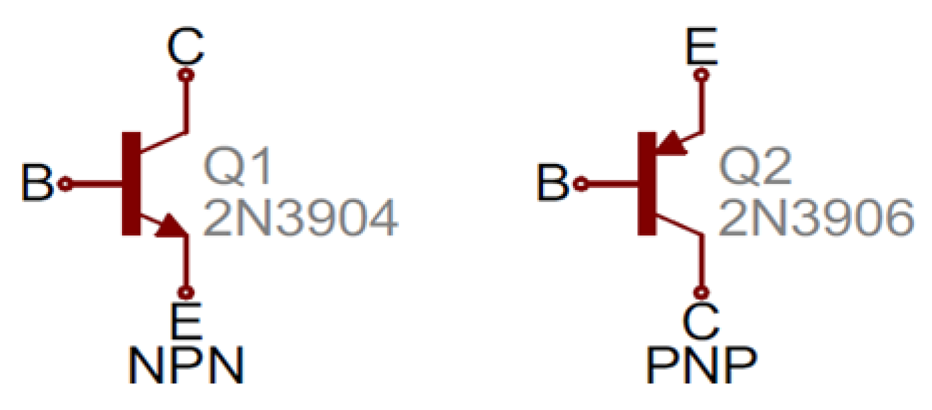


## الترانزستورات (Transistors)

يُمكن للترانزستورات -سواء كانت من نوع الترانزستور ثنائي القطبية (BJT) أو الموسفت (MOSFET)- أن تكون على شكلين: إما مطعمة تطعيماً موجباً (positively doped)أو مُطعمة تطعيماً سالباً .(negatively doped) لذلك كل نوع من تلك الأنواع له على الأقل طريقتين لتمثيله.

## الترانزستورات ثنائية القطبية (Bipolar Junction Transistors (BJTs)

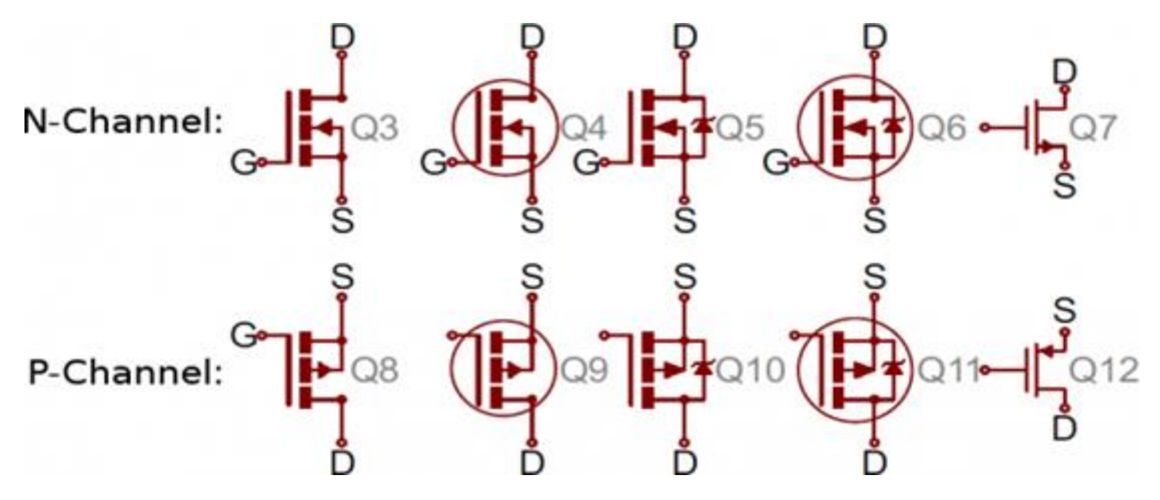
الترانزستورات ثنائية القطبية هي مكونات تحتوي على ثلاثة أطراف: مُجمع (collector (C))، وباعث(emitter (E)) ، وقاعدة .(base (B)) هناك نوعان من الترانزستورات ثنائية القطبية، NPNوPNP، وكل منهما له رمز خاص به.




المُجمع (C) والباعث (E) يكونان على خط واحد، لكن طرف الباعث يوجد به دائماً سهم. إذا كان السهم يُشير للداخل يكون الترانزستور من نوعPNP ، وإذا كان السهم يُشير للخارج يكون الترانزستور من نوع.NPN

**الموسفت (الترانزستور الحقلي المصنوع من أكسيد معدني) ( metal-oxide ((field-effect (MOSFET**

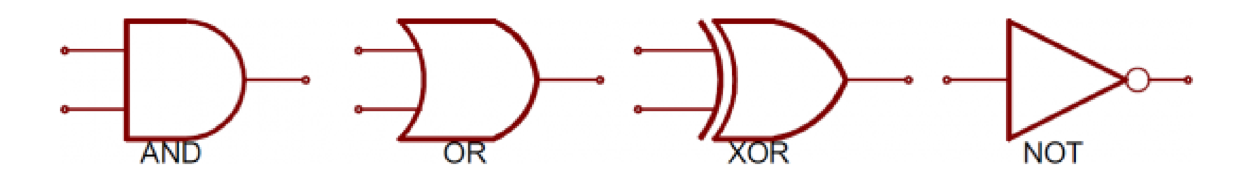
تحتوي الترانزستورات من النوع موسفت –كما هو الحال في ترانزستورات -BJT على ثلاثة أطراف، ولكن بأسماء مُختلفة؛ فأطراف هذه الترانزستورات تُسمى المصدر ((source (C)، المصرف((drain (D) ، والبوابة .((gate (G) أيضاً هنا نوعان مختلفان من الرموز يُستخدمان للإشارة إلى ترانزستوراتMOSFET ، فهناك نوعان من الموسفت، الأول n-channel والثاني .p-channel هناك عدد من الرموز يشيع استخدامها مع كلا نوعي الموسفت:



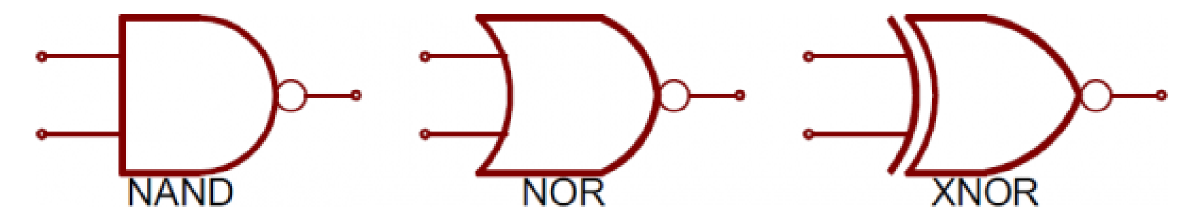

السهم الموجود في منتصف الرمز (يُسمى الحاجز ((bulk) يُحدد ما إذا كان الموسفت n-channelأو .p-channel إذا كان السهم يُشير للداخل فهذا يعني أن الموسفت من النوعيةn-channel ، وإذا كان السهم يُشير للخارج فهذا يعني أنه من النوعية-p channel.

# بوابات المنطق الرقمي (Digital Logic Gates)

بوابات المنطق الرقمي الأساسية التي نستخدمها AND و OR وNOT وXOR جميعها لها رموز تخطيطية خاصة بها:



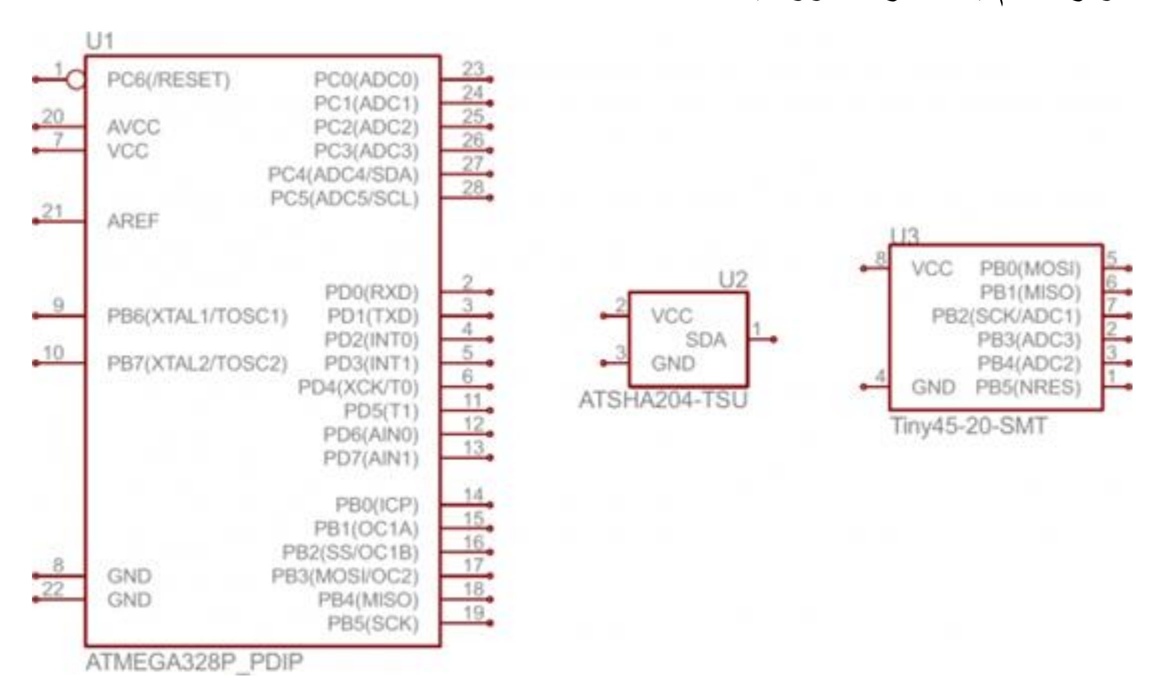

وبإضافة دائرة على جهة الخرج نقوم بعكس البوابات لتصبح NAND و NOR و:XNOR




من الممكن أن تحتوي رموز البوابات المنطقية على أكثر من طرفي دخل، ولكن أشكالها تبقى كما هي (مع إمكانية زيادة حجمها)، ولا بد كذلك من أن يكون هناك طرف خرج واحد فقط.

## الدوائر المتكاملة (Integrated Circuits)

تقوم الدوائر المتكاملة بمهام فريدة، لذلك فهي متنوعة بدرجة كبيرة مما لا يسمح بوجود رمز ثابت يتم استخدامه للإشارة إليها في مخططات الدوائر الإلكترونية. في الغالب يتم تمثيل الدوائر المتكاملة بمستطيل يحتوي على منافذ تبرز من جوانبه. وكل منفذ يُدون بجواره اسم يشمل رقمه ووظيفته.

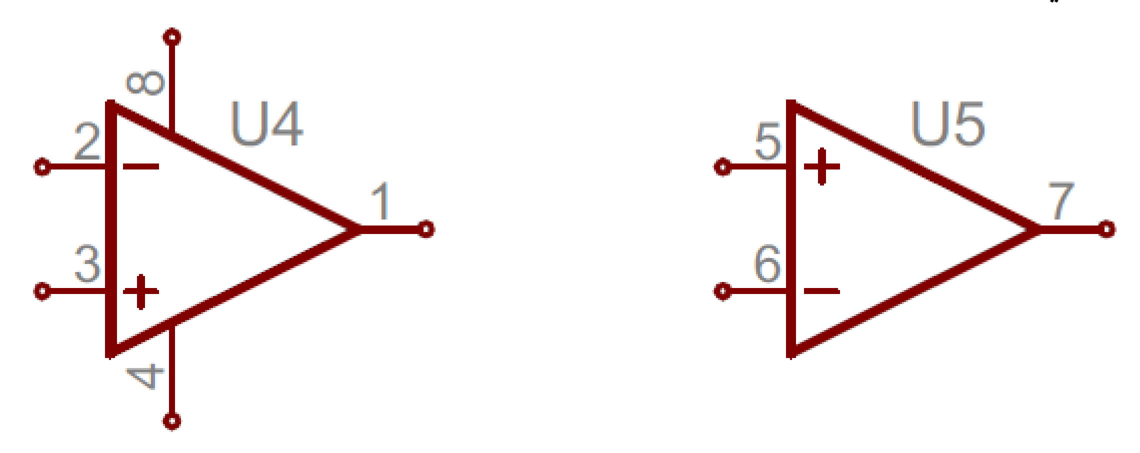



الرموز التخطيطية اعلاه الخاصة بمُتحكم دقيق ATmega328 (يوجد بكثرة في بطاقات أردوينو)، دائرة متكاملة للتشفير ATSHA204 ، ووحدة تحكم مصغرة ATtiny45 MCU. وكما ترى هذه المكونات تختلف بشكل كبير في الحجم وعدد المنافذ.

بسبب استخدام رمز عام مشترك للدوائر المتكاملة المختلفة تُصبح للأسماء والقيم التي يتم كتابتها أهمية كبيرة للغاية. وكل دائرة متكاملة يجب أن يُدون عليها قيم تحدد بدقة اسم الرقاقة.

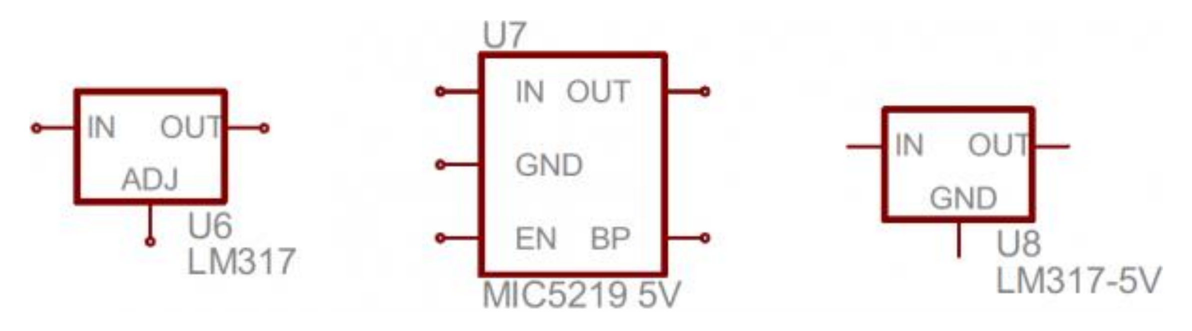
# المُضخمات العملياتية (Op Amps) ومُنظمات الجهد ( Voltage Regulators)

بعض الدوائر المتكاملة الأكثر استخداماً يكون لها رموز خاصة لتمثيلها في المُخططات. فالمضخمات العملياتية على سبيل المثال يتم تمثيلها بالرمز الموجود بالأسفل، حيث يحتوي على خمسة أطراف: طرف دخل غير معكوس (non-inverting input) موجب (+)، وطرف دخل معكوس (inverting input) سالب (-)، وطرف خرج وطرفي دخل للطاقة.



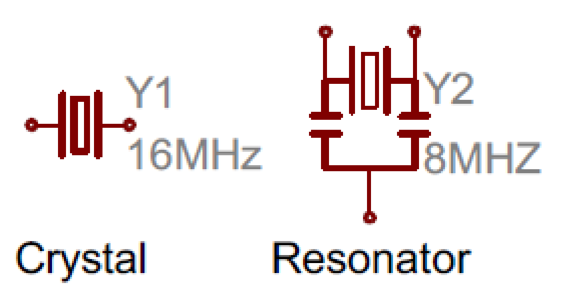

كثيراً ما يكون هناك مضخمان عملياتيان مدموجين معاً في دائرة متكاملة واحدة لا تتطلب سوى منفذ واحد للطاقة وآخر للأرضي. لهذا السبب يحتوي الرمز الموجود على اليمين على ثلاثة أطراف فقط.

مُنظمات الجهد البسيطة تحتوي في الغالب على ثلاثة أطراف، واحد للدخل وواحد للخرج وواحد للأرضي (أو للضبط). ويتم تمثيلها بمستطيل يحتوي على منافذ جهة اليسار (للدخل)، ومنافذ جهة اليمين (للخرج)، ومنافذ في الأسفل (للأرضي/ للضبط).



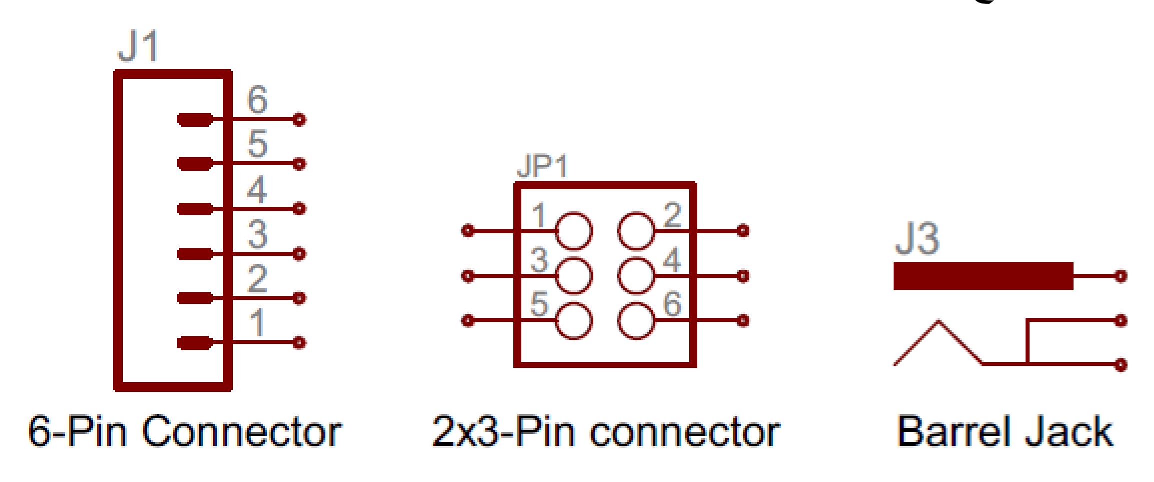
## الكريستالات (Crystals) والرنانات (Resonators)

تُشكل الكريستالات والرنانات جزءاً هاماً من دوائر المتحكمات الدقيقة (microcontroller). فهي تُساعد في توفير إشارة الساعة. رموز الكريستالات تحتوي غالباً على طرفين، بينما رموز الرنانات –التي هي في الأساس كريستالات يُضاف إليها مكثفان- تحتوي على ثلاثة أطراف.



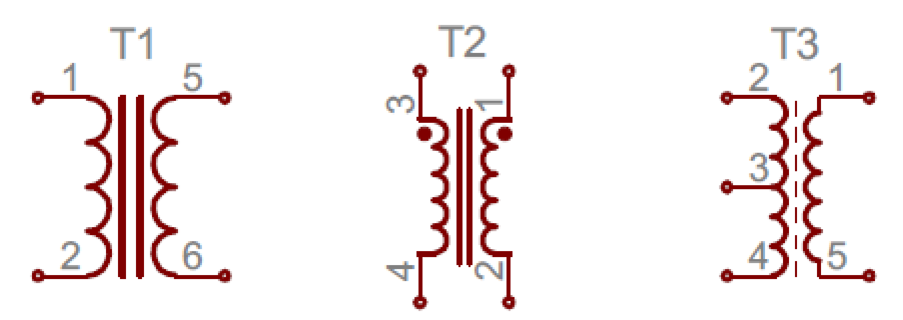

## الوصلات (Connectors) والرؤوس المُسننة (Headers)

نحتاج دائماً لاستخدام الوصلات في معظم الدوائر سواء كنت ترغب بتوصيل الطاقة أو إرسال المعلومات. والرموز المُستخدمة تختلف اعتماداً على شكل الوصلة، والأمثلة التالية توضح ذلك:

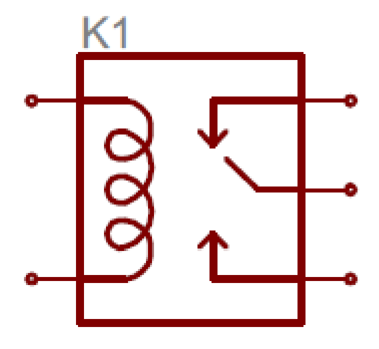


## المُحركات (Motors) والمُحولات (Transformers) ومُكبرات الصوت (Speakers) والمُرَحِّلات (Relays)

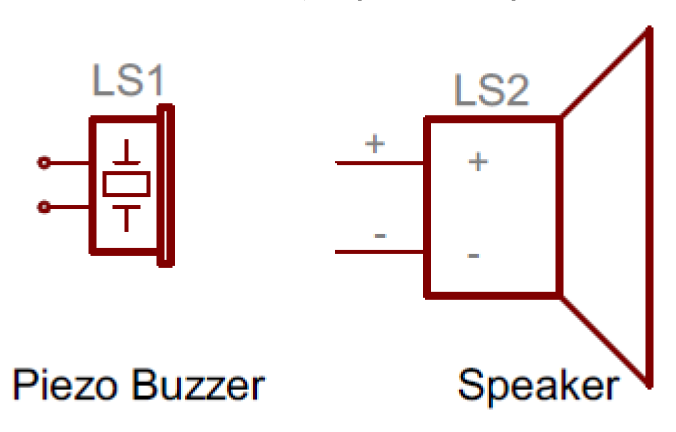
نذكر هذه المكونات معاً لأنها تشترك في احتوائها (غالباً) على لفات من الأسلاك (coils). فالمُحولات تحتوي على ملفين موضوعين مقابل بعضهما البعض مع وجود خطين يفصلان بينهما:



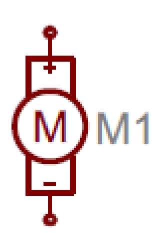

أما المُرحلات فيُرمز لها بملف مع مفتاح كما يلي:




مُكبرات الصوت والطنانات (buzzers) يتم تمثيلها برموز تُشبه شكلها في الواقع:

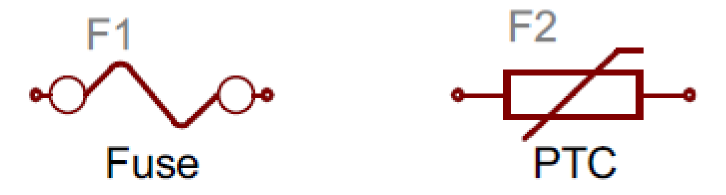


والمُحركات عموماً يتم تمثيلها بحرف "M" تحيط به دائرة مع بعض التعديل على شكل الطرفين:




**المُنصهرات (Fuses) والمُقاومات الحرارية (PTCs)**

كل من المُنصهرات والمُقاومات الحرارية –مُكونات تُستخدم للحد من الزيادات في شدة التيار الكهربي- يتم تمثيلها برمز خاص:




الرمز المُستخدم لتمثيل المُقاومات الحرارية هو الرمز العام المُستخدم لتمثيل الثرمستور thermistorلاحظ أن هذا الرمز هو الرمز المستخدم عالمياً للمُقاومات مع إضافة بسيطة).

لا شك أن هناك العديد من الرموز المُستخدمة في الدوائر الإلكترونية لم تشملها هذه القائمة، لكن الرموز التي ذكرناها تكفي بنسبة 90% لجعلك مُتمكناً من قراءة المُخططات. بشكل عام يكون هناك تشابه كبير بين المُكون في الواقع والرمز المُستخدم لتمثيله في المُخططات. وبالإضافة للرموز لا بد أن يكون لكل مُكون في المُخططات اسم وقيمة للمساعدة بشكل أكبر في التعرف عليه.

أحد أهم المفاتيح لكي تُصبح مُلماً بمُخططات الدوائر الإلكترونية معرفة كيفية التفريق بين المكونات. رموز المكونات تُخبرنا نصف المعلومات، ولكن كل رمز لا بد أن يصحبه اسم وقيمة لكي نحصل على المعلومة كاملة.

تُساعد القيم على تحديد ماهية المكون بدقة. بالنسبة للرموز التخطيطية لمكونات مثل المُقاومات والمُكثفات وملفات الحث تخبرنا القيمة عن القيمة التي يحتويها المكون بالأوم أو الفاراد أو الهنري. أما بالنسبة للمكونات الأخرى –مثل الدوائر المتكاملة- من الممكن أن تكون القيمة اسم الرقاقة فحسب. وفي الكريستالات من المُمكن أن تكون القيمة هي تردد التذبذب الخاص بها. إذن بشكل أساسي تذكر القيمة الخاصة بالرموز التخطيطية الخاصية الأهم للمكون.

تتكون أسماء المكونات في الغالب من حرف أو حرفين مع رقم. الجزء الحرفي يحدد نوع المكون R – للإشارة إلى المُقاومات، Cللمُكثفات، Uللدوائر المُتكاملة... الخ. ولا بد من أن يُعطى كل مُكون على المُخطط اسماً فريداً؛ فمثلاً إذا كان هناك عدة مُقاومات في مُخطط ما يتم تسميتهاR1 ، R2، ...R3الخ.

بادئات الأسماء موحدة وقياسية. ففي بعض المكونات –مثل المقاومات- تكون البادئة الحرف الأول من اسم المكون (في حالة المقاومات R). أما بعض المكونات الأخرى لا تكون البادئة مُشتقة من اسمها؛ بادئة ملفات الحث على سبيل المثال L (لأن الحرف I يتم استخدامه للدلالة على شدة التيار، وبالتالي قد يحدث تعارض والتباس)، إليك جدول مُبسط يحتوي على المُكونات الأساسية وبادئات أسمائها:

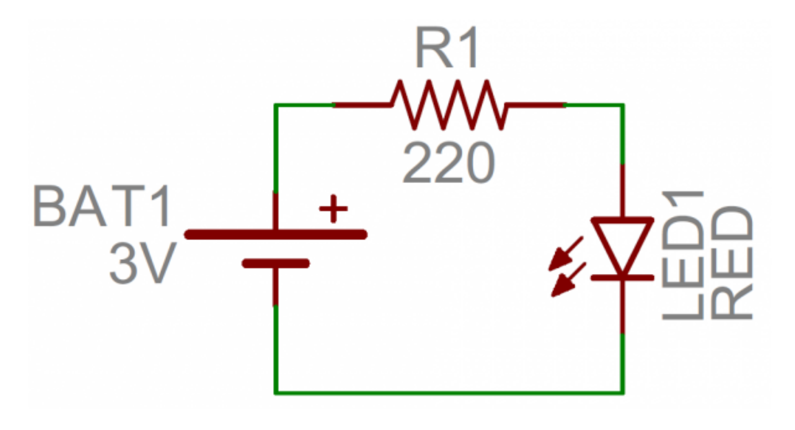
| بادئة الاسم | المُكون |
|---|---|
| R | المُقاومات |
| C | المُكثفات |
| L | ملفات الحث |
| S | المفاتيح |
| D | الديودات |
| Q | الترانزستورات |
| U | الدوائر المُتكاملة |
| Y | الكريستالات والرنانات |

برغم أن تلك الأسماء الخاصة برموز المُكونات "قياسية" إلا أنها لا يتم اتباعها بشكل دائم من الجميع. فعلى سبيل المثال من المُمكن أن تُستخدم البادئة IC مع الدوائر المتكاملة بدلاً من U، وكذلك من الممكن استخدام XTAL مع الكريستالات بدلاً من Y. حاول بأقصى قدرة لديك التفريق بين رموز المكونات وفهمها بشكل الصحيح. ويجب أن يحتوي كل رمز على معلومات كافية للتعرف على المكون الذي يمثله.

فهم كيفية التفريق بين المُكونات الموجودة على المُخططات يمثل ما يزيد على نصف الطريق نحو فهم المُخططات. ويتبقى الآن تحديد كيفية اتصال جميع رموز المُكونات مع بعضها البعض.

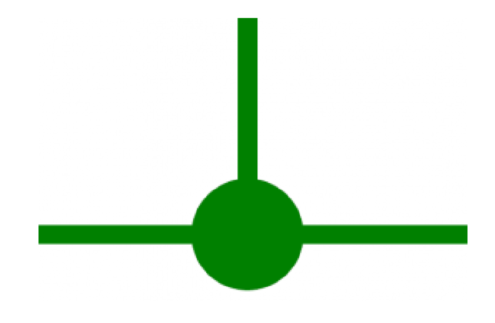
## الشبكات (Nets) والعُقَد (Nodes) والتسميات (Labels)

تُخبرنا شبكات المُخططات كيفية اتصال المُكونات ببعضها البعض بداخل الدائرة الكهربية. يتم تمثيل الشبكات بخطوط تصل بين أطراف المُكونات. وأحياناً (وليس دائماً) يتم منح الشبكات لوناً خاصاً بها، مثل الخطوط الخضراء الموجودة في هذا المُخطط:

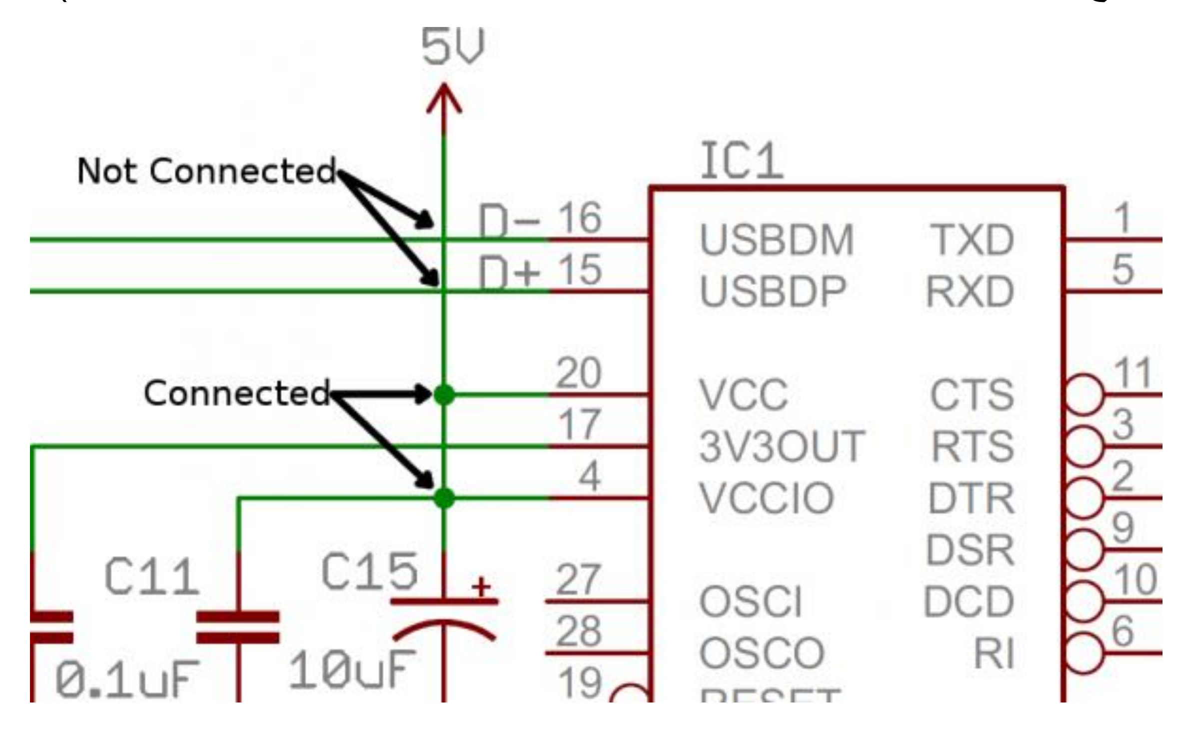



## التقاطعات (Junctions) والعُقد

من الممكن استخدام الأسلاك لتوصيل طرفين ببعضهما البعض، وكذلك من الممكن استخدامها لتوصيل الكثير من الأطراف معاً. عندما ينقسم سلك ما في اتجاهين ينتج عن ذلك تقاطع. وفي المُخططات يتم تمثيل التقاطعات باستخدام العقد، وهي عبارة عن نقاط صغيرة توضع في نقاط تقاطع الأسلاك.

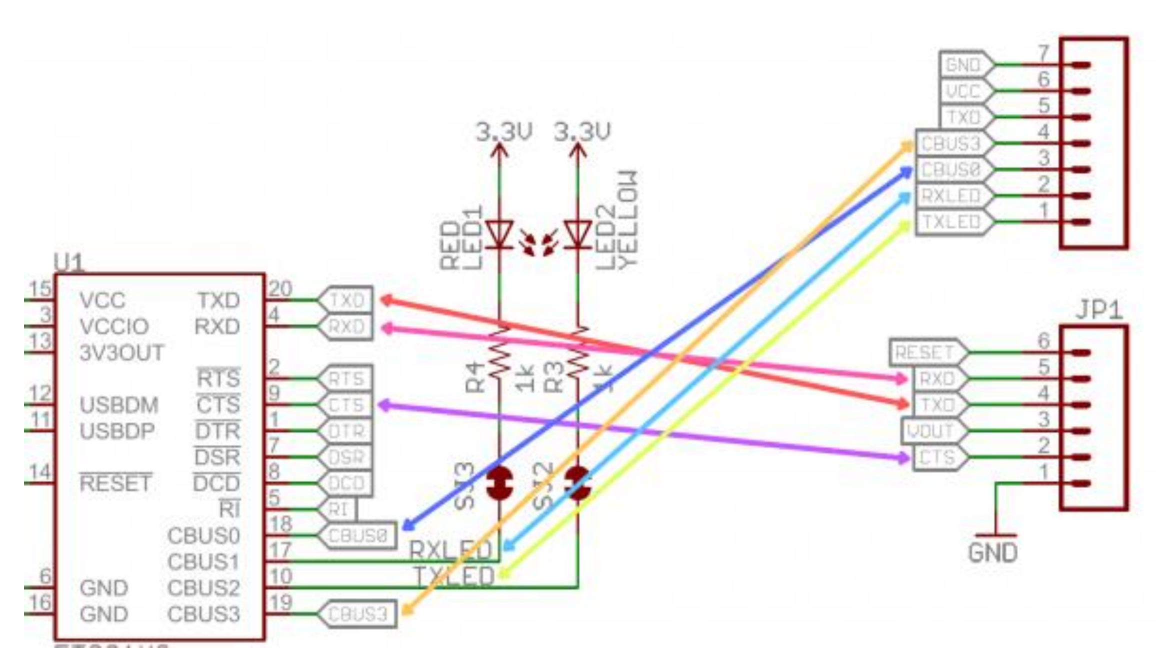


تمنحنا العُقد القدرة على معرفة أن الأسلاك المارة بنقطة تقاطع ما متصلة مع بعضها. فعدم وجود عُقدة عند نقطة تقاطع ما يعني أن الأسلاك المارة بتلك النقطة ليست مُتصلة ببعضها، وإنما تمر من خلالها فحسب. (عند تصميم المُخططات من الجيد أن تتجنب تقاطع الأسلاك دون اتصال إلى أقصى درجة ممكنة، لكن أحياناً يستحيل تجنب ذلك).



**أسماء خطوط الشبكات**

أحياناً لتسهيل قراءة المُخططات نقوم بإعطاء أسماء لخطوط الشبكات وكتابتها عليها، بدلاً من توصيل الأسلاك في المُخطط. خطوط الشبكات التي تحمل نفس الاسم يُفترض أنها تكون مُتصلة معاً حتى لو لم تكن هناك أسلاك تصل بينها على المُخطط. من المُمكن كتابة الأسماء مُباشرة فوق خطوط الشبكة أو من الممكن أن تُكتب مُتدلية من الخطوط.

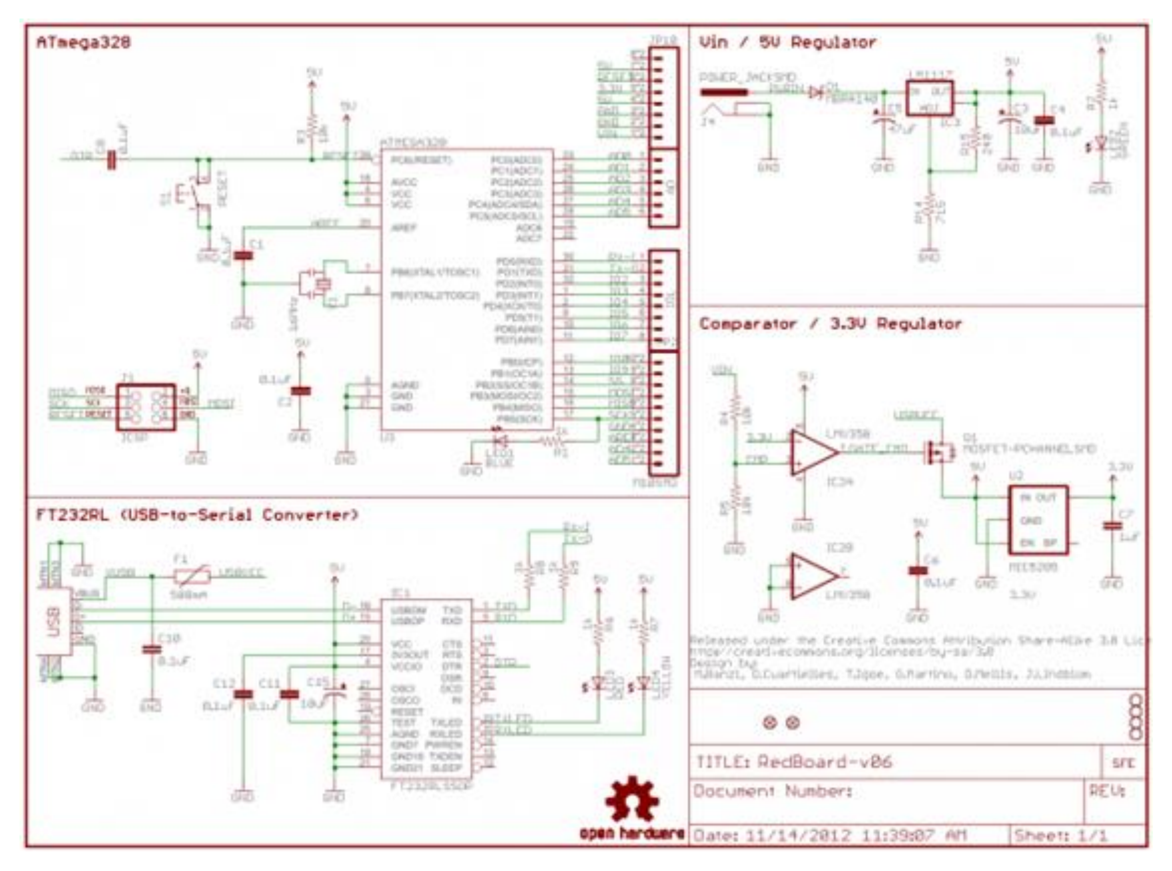



جميع خطوط الشبكات التي تحمل نفس الاسم تكون مُتصلة معاً، ومثال على ذلك هذا المُخطط الخاص بلوح FT231X Breakout Board. استخدام الأسماء يحد من الفوضى في المُخططات (تخيل أنه تم توصيل جميع خطوط هذا المخطط.

عادة تُمنح خطوط الشبكات أسماء تدل على الغرض من الإشارات التي تحملها الأسلاك التي تمثلها تلك الشبكات. على سبيل المثال خطوط الطاقة تُسمى “VCC” أو “5V”، بينما شبكات الاتصال التسلسلي من الممكن تسميتها “RX” أو “TX”.

## نصائح لقراءة المخططات

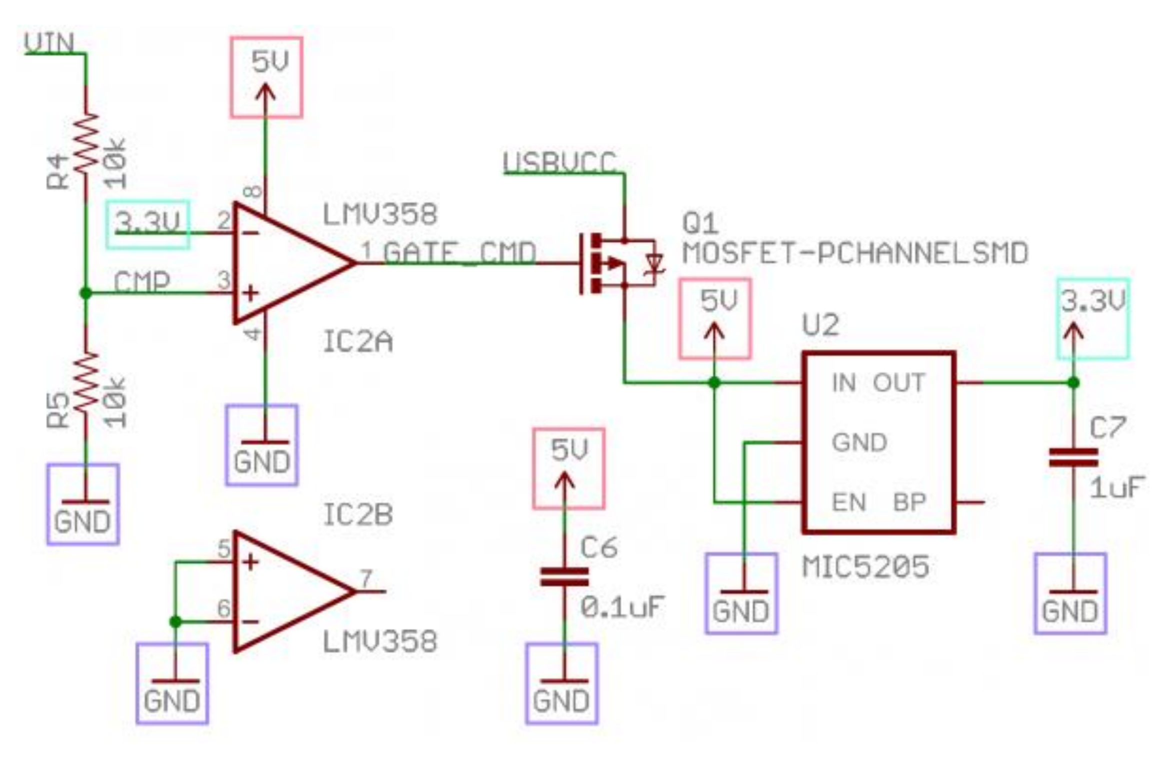
من الضروري أن يتم تقسيم المُخططات الكبيرة للغاية إلى قوالب وظيفية. من الممكن أن يكون هناك جزءاً خاصاً بدخل الطاقة وبتنظيم الجهد، أو جزء خاص بمُتحكم دقيق، أو قسم مُخصص للوصلات. حاول التفريق بين الأجزاء المختلفة واتباع سير الدائرة من الدخل إلى الخرج. مُصممو مُخططات الدوائر المتميزين من الممكن أن يقوموا بجعل الدائرة مثل الكتاب، بحيث تكون المُدخلات على الجانب الأيسر والمُخرجات على الجانب الأيمن.




إذا كان راسم المُخطط مُحترفاً (مثل المهندس الذي قام بتصميم المُخطط الخاص بلوح RedBoard) فإنه يقوم بالفصل بين أجزاء المُخطط إلى قوالب مع تسمية كل منها.

# التعرف على عُقد الجهد

عُقد الجهد هي عبارة عن مكونات أحادية الطرف على المُخططات، وهي التي يتم توصيل أطراف المُكونات الأخرى بها لجعلها عند مستوى جهد مُعين. وهذا من تطبيقات الخاصة بأسماء خطوط الشبكات، بمعنى أن جميع الأطراف المُتصلة بُعقد جهد لها نفس الاسم تكون متصلة ببعضها البعض.




عُقد الجهد التي تحمل نفس الاسم –مثلGND ، V5، 3.3 -Vتكون مُتصلة مع بعضها البعض بالرغم من عدم رسم أسلاك تصل بينها على المُخطط.

عُقد الجهد الأرضي لها أهمية خاصة لأن هناك الكثير من المُكونات التي تكون بحاجة للتوصيل بالأرضي.

## واخيرا الرجوع إلى صحف بيانات (Datasheets) المُكونات

إذا وجدت شيئاً ما لا تفهمه على مُخطط ما فحاول الحصول على صحيفة البيانات الخاصة بأكثر المُكونات أهمية. وعادة ما يكون المكون الأهم والذي يقوم بمعظم العمل عبارة عن دائرة متكاملة مثل متحكم دقيق أو مستشعر (sensor). وعادة ما يكون كذلك المكون الأكبر، ومن المحتمل أيضاً أن يكون في منتصف المُخطط.




Arduino هو كومبيوتر صغير الحجم بإمكانه التفاعل و التحكم في الوسط المحيط به بشكل أفضل من الكومبيوتر المكتبي Desktop. تقنيا هو منصة Platform (البيئة التي يتم فيها تشغيل البرمجيات) برمجية مفتوحة المصدر تتكون من متحكم إلكتروني Micro-Controller و بيئة تطويرية تكاملية لكتابة البرامج والاوامر( Integrated IDE (Development Environment.

قوة الآردوينو Arduino تتجلى في قدرته الكبيرة على تصميم المشاريع الإلكترونية التفاعلية والتواصل مع القطع الإلكترونية الأخرى كالمحولات أو المستشعرات Sensors و الاستفادة منها في الحصول على مختلف البيانات كدرجة الحرارة أو شدة الإضاءة و كذلك فاعليته الكبيرة في التحكم في المحركات Motors و مصابيح LED و كثير من القطع الإلكترونية الأخرى التي سبق درسناها في الجزء الثاني من السلسلة.

يمكن تشغيل مشاريع الآردوينو Arduino عن طريق وصله USB بالكومبيوتر و جعله يتعامل مع أحد البرامج الموجودة على الجهاز أو بالإمكان تشغيله باستقلالية تامة.

بالإضافة لذلك تستخدم بيئة التطوير المتكاملة الخاصة بالأردوينو نسخة مبسطة من لغة C++ والتي سبق درسناها في الجزء الثالث من السلسلة مما يسهل تعلم عملية البرمجة وهي لغة مفتوحة المصدر .

ظهرت فكرة جهاز الآردوينو عام 2005 في مدينة إيفريا الإيطالية، حيث أطلق ماسيمو بانزى بالتعاون مع دايفيد كوارتيليس وجاينلوكا مارتينو بإطلاق مشروع "أردوين إيفريا" (Arduin of Ivrea) وسُمّيّ المشروع باسم أشهر شخصية تاريخية في المدينة. وكان الهدف الأساسي للمشروع هو عمل بيئة تطوير للمتحكمات دقيقه بصوره مفتوحه المصدر 100 في المئة وتضمن هذا المشروع عمل بيئة تطوير برمجيه للمتحكمات الدقيقة Integrated Development Environment وتكون مجانيه في ذات الوقت كما تضمن عمل لوحات تطوير Development Boards صغيره الحجم بتكلفه بسيطه تبلغ حالياً قرابة 10 دولار واقل ليتمكن الطلاب والهواة التقنين تحمل سعرها، وحتى عام 2013 تم شحن أكثر من 700 ألف لوحة آردوينو.

ما يميز الآردوينو Arduino هو مجموعة من الأمور اللتي تصنع الفارق بينه و بين غيره أهمها:

البساطة: قطعة الآردوينو Arduino مصممة لتناسب احتياجات الجميع، محترفين، أساتذة، طلاب وهواة الإلكترونيات التفاعلية.

الثمـــن: لوح الآردوينو Arduino أقل ثمناً مقارنةً مع الألواح الأخرى من نفس النوع فثمن أغلى Arduino لا يتجاوز 100$

التركيب الذاتي :(Self-Assembly) يمكنك تحميل ورقة البيانات Datasheet الخاصة بالآردوينو Arduino مجاناً من الموقع الرسمي و شراء القطع وتركيبه بنفسك!

متعدد المنصات: برنامج الآردوينو له القدرة على الاشتغال على الويندوز windows, الماك Mac OS و اللينكس Linux وأغلب المتحكمات الإلكترونية الأخرى تشتغل فقط على الويندوز فقط.

بيئة برمجية سهلة و بسيطة: البيئة البرمجية Programming Environment مصممة لتكون سهلة للمبتدئين و ثابتة و قوية للمحترفين.

مفتوح المصدر Open Source Software مكتوب بلغة السي C++ و متاح للجميع لتحميله و بامكان المبرمجين التعديل عليه وفق احتياجاتهم.

Open Source Hardware: الآردوينو Arduino مصنوع أساساً من متحكمات ATMEGA8 و ATMEGA168 و المخططات منشورة تحت ترخيص

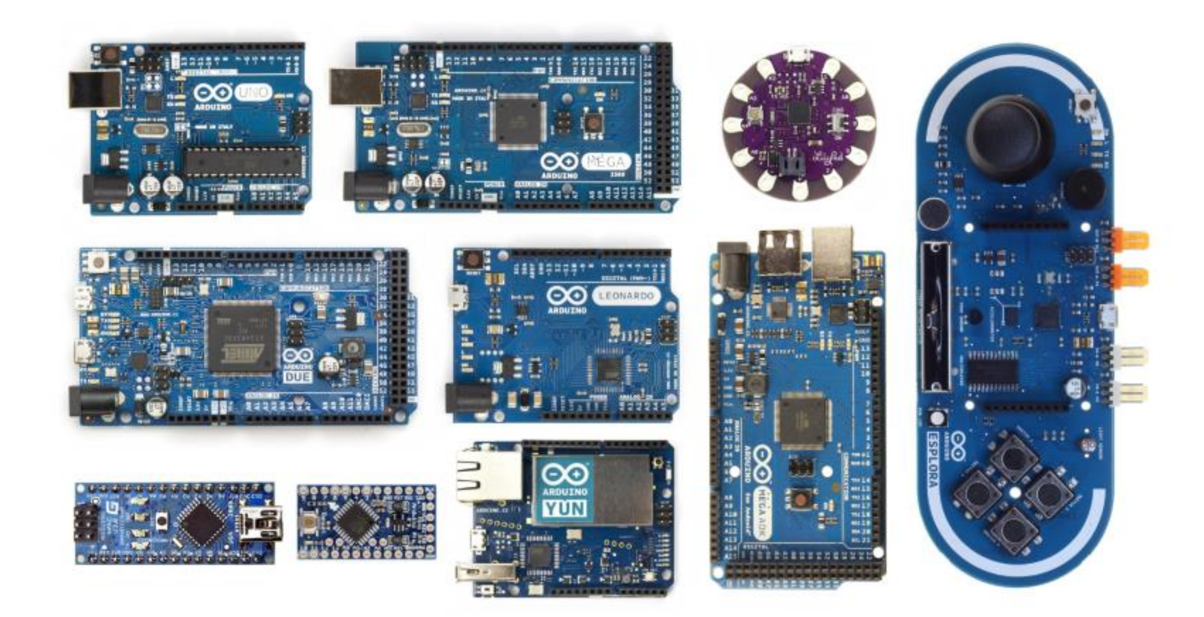
Creative Commons مما يتيح إلى مصممي الدارات الإلكترونية Electronic Circuits تصميم داراتهم الخاصة.

ماذا نقصد بمفتوح المصدر ؟

مخططات تصميم العتاد Hardware Schema الخاصة بالآردوينو Arduino متاحة للجميع لتحميلها ودراستها لفهم مبدأ عمل القطعة و التعديل عليها و كذلك إمكانية الإستفادة منها تجارياً وهذا وفقاً لبنود اتفاقية.Creative commons

كذلك الكود المصدري الخاص ببرنامج Arduino مفتوح المصدر ومتوفر بترخيص GPL

# انواع الاردوينو

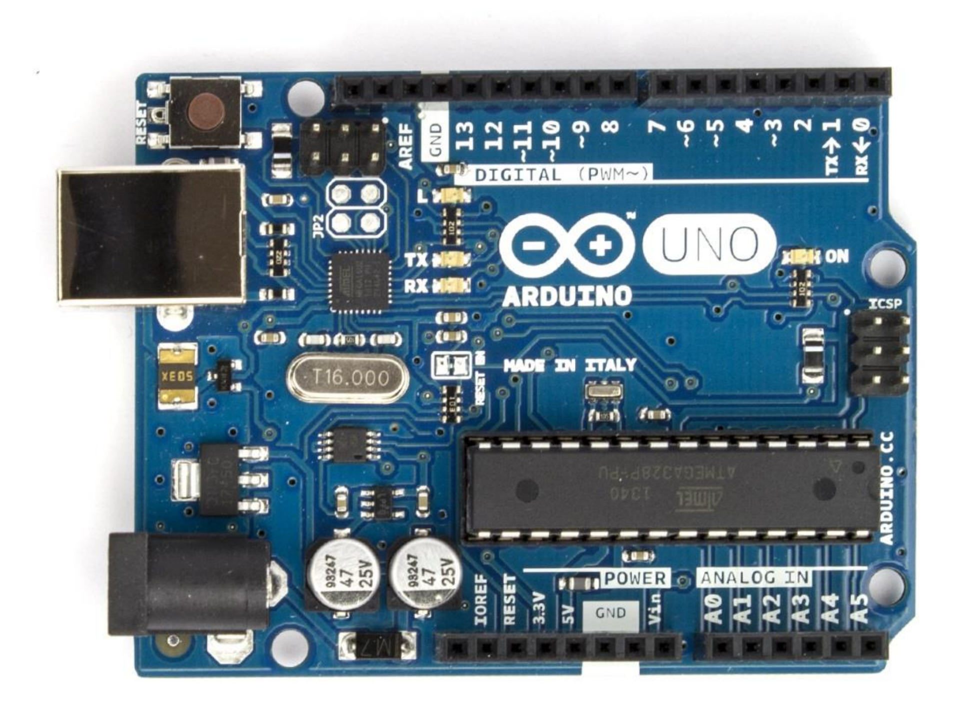



يوجد للاردوينو انواع مختلفة لتتناسب احتياجات المشاريع التي يعمل بها المستخدم وابرز النقاط التي تختلف في الانواع هي ما يلي :

1- عدد المخارج والمداخل الرقمية والتناظرية والتي تحدد عدد الاجهزة التي يمكن التحكم بها وعدد الحساسات التي يمكن دمجها .
2- نوع المتحكم المستخدم .
3- سرعة المعالج الموجود بداخلها وامكانية تبديلها او لا
4- حجم ووزن الاردوينو , الابعاد الهندسية (الطول والعرض والسمك) والوزن /غم
5- الجهد والتيار التشغيلي
6- جهد تغذية النظام
7- حجم الذاكرة
8- السعر
9- الشكل , مربع او مستطيل او دائري
10- نوع المنفذ التسلسلي للبرمجة او التغذية

من الانواع المتوفرة هي أردوينو أونو (Arduino Uno)

الأردوينو أونو يعتبر من أشهر أنواع الأردوينو وأكثرها استخداماً, يستخدم الأردوينو أونو متحكم أصغري من نوع ATmega328 تحتوي لوحة الاردوينو اونو على 14 منفذ رقمي Degital (إدخال/إخراج) (6 يمكن استخدامهم كمنافذ للتحكم بالتماثل العرضي للنبضة – PWM outputs) وهي التي يوجد على جانبها أشارة " ~ "). توفر أيضاً لوحة الاونو أيضاً 6 منافذ تماثلية-تناظرية (analog input). بالإضافة إلى منفذ لوصلة (USB) والتي تستطيع من خلاها تزويد الأردوينو بالطاقة وتحميل نص البرمجة على المتحكمة الأصغرية, ومنفذ آخر لتزويد الاردوينو اونو بطاقة خارجية منفصلة (مثل البطاريات ذات ال 9 فولت), كما تحتوي لوحة الاردوينو اونو على زر لإعادة التشغيل و رؤوس قابلة للتوصيل تسمح للمستخدم باستخدام برمجة بالتتابع (ICSP HEADER).

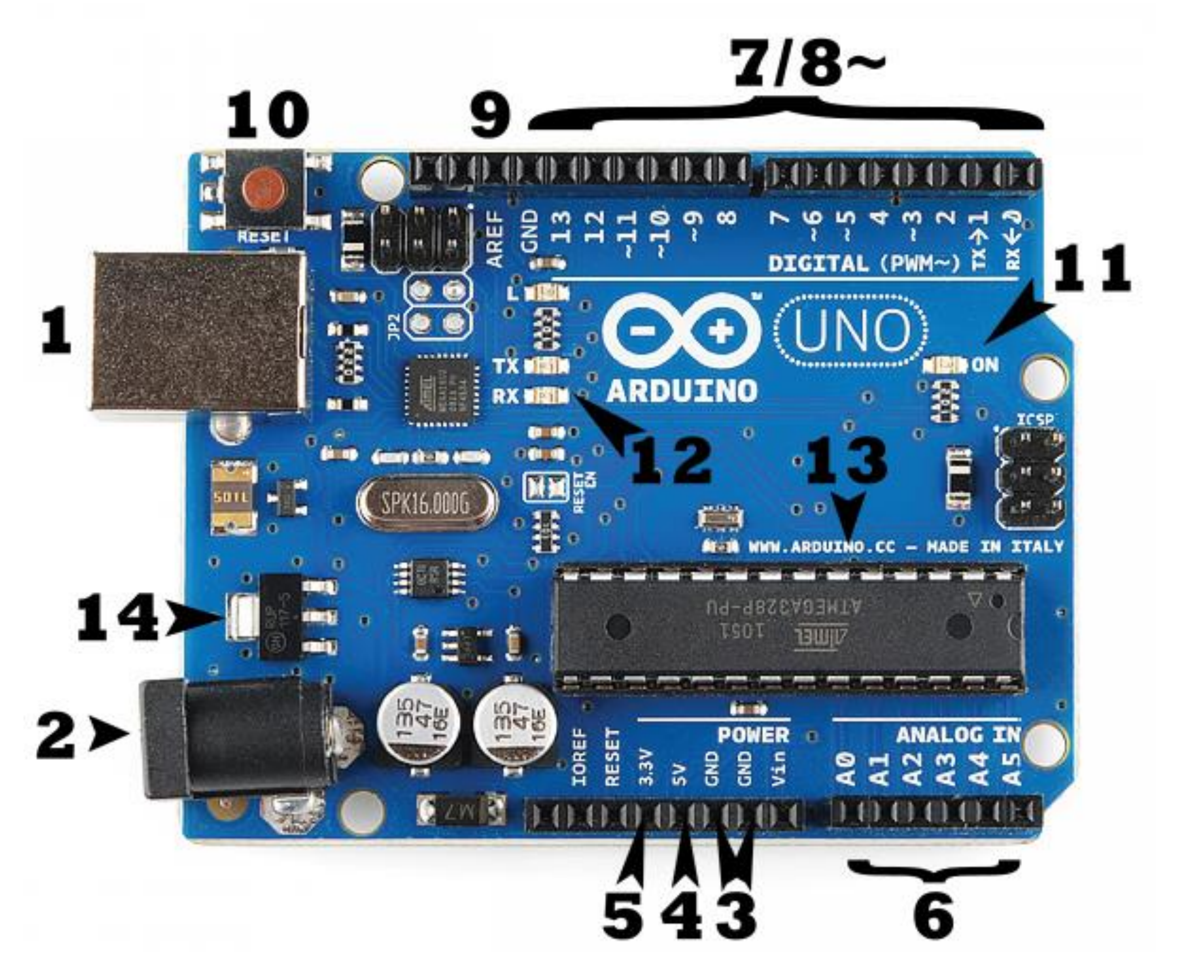


الملخص:

| | |
|---|---|
| •المتحكم الأصغري: | ATmega328 |
| •جهد تشغيل النظام الكهربائي: | 5 v |
| •فولطية المنفذ (الموصى به): | 7- 12 v |
| •فولطية المنفذ (الحد الأقصى والأدني): | 6- 20 v |
| •عدد المنافذ الرقمية (إدخال/إخراج): | 14 (6 يمكن استخدامهم كمنافذ للتحكم بالتماثل العرضي للنبضة) |
| •عدد المنافذ التماثلية (إدخال): | 6 |
| •التيار المستمر لمنفذ 3.3 فولت: | 50mA |
| •التيار المستمر لمنفذ (إدخال/إخراج) رقمي: | 40mA |
| •سرعة المعالج: | 16MHz |
| •حجم الذاكرة: | 32 كيلو بايت (0.5 كيلو بايت تستخدم لمحمِّل الإقلاع) |
| •أبعاد اللوحة: | الطول: 2.7 إنش , العرض: 2.1 إنش |
| •السعر: | $29.96 |

للتعرف على جميع انواع الاردوينو المتوفرة انظر الى الملحق A في نهاية الكتاب.

مخطط محتويات اردوينو اونو للتعرف على بعض المكونات الاساسية للاردوينو (UNO R3)




**مدخل الطاقة (مقبس اسطواني/ USB)**

تحتاج جميع ألواح الأردوينو لطريقة ما ليتم توصيلها بمصدر الطاقة. أردوينو أونو يمكن إمداده بالطاقة عن طريق كابل USB من الكمبيوتر أو من مأخذ الحائط باستخدام مقبس اسطواني (Barrel Jack). في الصورة بالأعلى يشير الرقم (1) لوصلة USB بينما يشير رقم (2) للمقبس الاسطواني.

يتم أيضاً من خلال وصلة USB تحميل الكود البرمجي إلى لوح الأردوينو. لمعرفة المزيد عن كيفية برمجة الأردوينو يمنك قراءة هذا الدرس تنصيب وبرمجة الأردوينو (قريباً).

**ملحوظة**: لا تستخدم أي مزود طاقة أكبر من 20 فولت لأن الأردوينو لا يتحمل جهد أكبر من ذلك، وسيتم تدميره. الجهد المُوصى به لمعظم إصدارات الأردوينو هو بين 6 و 12 فولت.

### المنافذ (pins) (V5، V3.3، أرضي (GND)، تناظري (Analog)، رقمي (Digital)، تعديل عرض النبضة (PWM)، المرجع التناظري (AREF)).

المنافذ الموجودة على لوح الأردوينو هي الأماكن التي تقوم بتوصيل الأسلاك بها لبناء دائرة الكترونية (مع استخدام لوح تجارب (breadboard) والأسلاك).
تحتوي تلك المنافذ غالباً على "رؤوس" سوداء بلاستيكية تسمح لك بتوصيل سلك إلى اللوح. توجد أنواع مختلفة من المنافذ، كل منها يتم كتابة اسم أو رمز مجاور له على اللوح للتفرقة بينهم ولها العديد من الوظائف المختلفة.

- GND (3): اختصار لكلمة أرضي (ground). توجد العديد من منافذ GND في لوح الأردوينو، يمكن استخدام أي منها لتأريض الدائرة.
- V (45) و V(53): هذان المنفذان لتزويد المكونات التي تُوصل بالأردوينو بالطاقة. منفذ V5 يزود بـ 5 فولت، بينما منفذ V3.3 يزود بـ 3.3 فولت. معظم المكونات البسيطة التي تُستخدم مع الأردوينو تعمل بشكل جيد بجهد 5 أو 3.3 فولت.
- Analog (6): المنافذ الموجودة في المنطقة أسفل كلمة (Analog In) (من A0 إلى A5 في أردوينو أونو) هي منافذ الدخل التناظري. هذه المنافذ يمكنها قراءة الإشارة القادمة من المستشعرات التناظرية (مثل مستشعر الحرارة) ثم تحويلها إلى قيمة رقمية يمكننا قراءتها.
- Digital (7): على الجانب الآخر من المنافذ التناظرية توجد المنافذ الرقمية (من 0 إلى 13 في أردوينو أونو). هذه المنافذ يمكن استخدامها لكل من الدخل الرقمي (مثل إخبارك إذا تم ضغط زر ما) والخرج الرقمي (مثل إضاءة ديود ضوئي).
- PWM (8): ربما لاحظت العلامة (~) بجوار بعض المنافذ الرقمية (3، 5، 6، 9، 10، 11 في أردوينو أونو). هذه المنافذ يمكنها العمل كمنافذ رقمية عادية، ولكن يمكن أيضاً استخدامها في شيء يسمى تعديل عرض النبضة (-Pulse Width Modulation). هناك درس عن تعديل عرض النبضة (قريباً)، ولكن الآن تخيل أن هذه المنافذ يمكنها محاكاة منفذ خرج تناظري.

- AREF (9): اختصاراً لكلمة "مرجع تناظري" (Analog Reference). في معظم الأحيان يتم ترك هذا المنفذ بدون أي وظيفة. لكن في بعض الأحيان يتم استخدامه لتعيين جهد خارجي مرجعي (بين 0 و5 فولت) كحد أقصى لمنافذ الدخل التناظري.

**زر إعادة الضبط (10) (Reset Button)**

تماماً مثل جهاز نينتندو (original Nintendo) يحتوي الأردوينو على زر إعادة ضبط (10). عند ضغط ذلك الزر يتم توصيل منفذ إعادة الضبط بالأرضي بشكل مؤقت وحذف أي كود محمل على الأردوينو ويتم استعادة برنامج التشغيل الأصلي للجهاز (الذي كان عليه وقت شراءه). قد يكون هذا مفيداً في حال كون الكود الخاص بك لا يتكرر، لكن عليك أن تجربه عدة مرات. على عكس نينتندو لا يساعد النفخ في الأردوينو في حل أي مشاكل.

**مؤشر ديود ضوئي للطاقة (11) (Power LED Indicator)**

أسفل يمين كلمة "أونو" الموجودة على لوح الأردوينو يوجد ديود ضوئي صغير مجاوراً لكلمة "ON" (مشار له في الصورة بالرقم 11). هذا الديود الضوئي يضيء عندما تقوم بتوصيل الأردوينو بمصدر للطاقة. إذا لم يعمل هذا الضوء، فمن المحتمل أن تكون هناك مشكلة، ويجب عليك أن تفحص الدائرة التي قمت بتركيبها.

**ديودات ضوئية TX و RX (TX RX LEDs)**

TX هو اختصار لكلمة إرسال ((transmit، RX هو اختصار لكلمة استقبال (receive). هذه العلامات توجد في الإلكترونيات للدلالة على المنافذ المسئولة عن الاتصال التسلسلي (serial communication). في حالتنا يوجد مكانين في أردوينو أونو يظهر فيهما TX و RX- الأول بجوار المنافذ الرقمية 0 و 1، والمكان الثاني بجوار ديودات مؤشرات الإرسال والاستقبال المضيئة (12). هذه الديودات تعطي لنا مؤشراً مناسباً لمعرفة إذا كان الأردوينو يرسل أو يستقبل البيانات (مثل أثناء تحميل برنامج جديد إلى لوح أردوينو).

**الدائرة المتكاملة الرئيسية (main IC)**

الشيء الأسود الذي يحتوي على أرجل معدنية (13) هو عبارة عن دائرة متكاملة (integrated circuit). فكر في الأمر كأن هذه الدائرة المتكاملة هي مخ

الأردوينو. يختلف نوع الدائرة المتكاملة المتواجدة في الأردوينو باختلاف اللوح، ولكن غالباً تكون من خط ATmega لإنتاج الدوائر المتكاملة التابع لشركة ATMEL.

من المهم أن تعرف نوع الدائرة المتكاملة (مع معرفة نوع لوح الأردوينو) قبل تحميل أي برنامج جديد إلى الأردوينو. هذه المعلومات يمكن أن توجد مكتوبة على الجانب العلوي من الدائرة المتكاملة. إذا أردت معرفة المزيد عن الفرق بين الدوائر المتكاملة المختلفة ربما يفيدك قراءة صحف البيانات المرفقة مع الألواح.

**منظم الجهد (voltage regulator)**

منظم الجهد (14) هو شيء لا يمكنك (ولا يجب عليك) التعامل معه على الأردوينو. ولكن فائدته هي تعريفنا أنه موجود ووظيفته التي يقوم بها. منظم الجهد يقوم تماماً بما يُفهم من اسمه- يقوم بالتحكم في كمية الجهد التي تسري في لوح الأردوينو. فكر في الأمر وكأنه بواب؛ ووظيفته هي تصريف أي جهد فائض يمكن أن يضر الدائرة. لكن بالطبع لديه حدود، لذلك لا تقم بتوصيل الأردوينو بأي مصدر جهد أعلى من 20 فولت.

# تجهيزات ادوات الاردوينو الاساسية

سوف نستخدم في هذه السلسلة الاردوينو اونو بشكل اساسي بخلاف الانواع الاخرى لانه متوفرا في الاسواق وسعره جيد وامكانية تطبيق المشاريه باستخدامه كبيرة جدا وحماسية , يحتوي على عدد اطراف يتناسب مع احتياجاتنا .

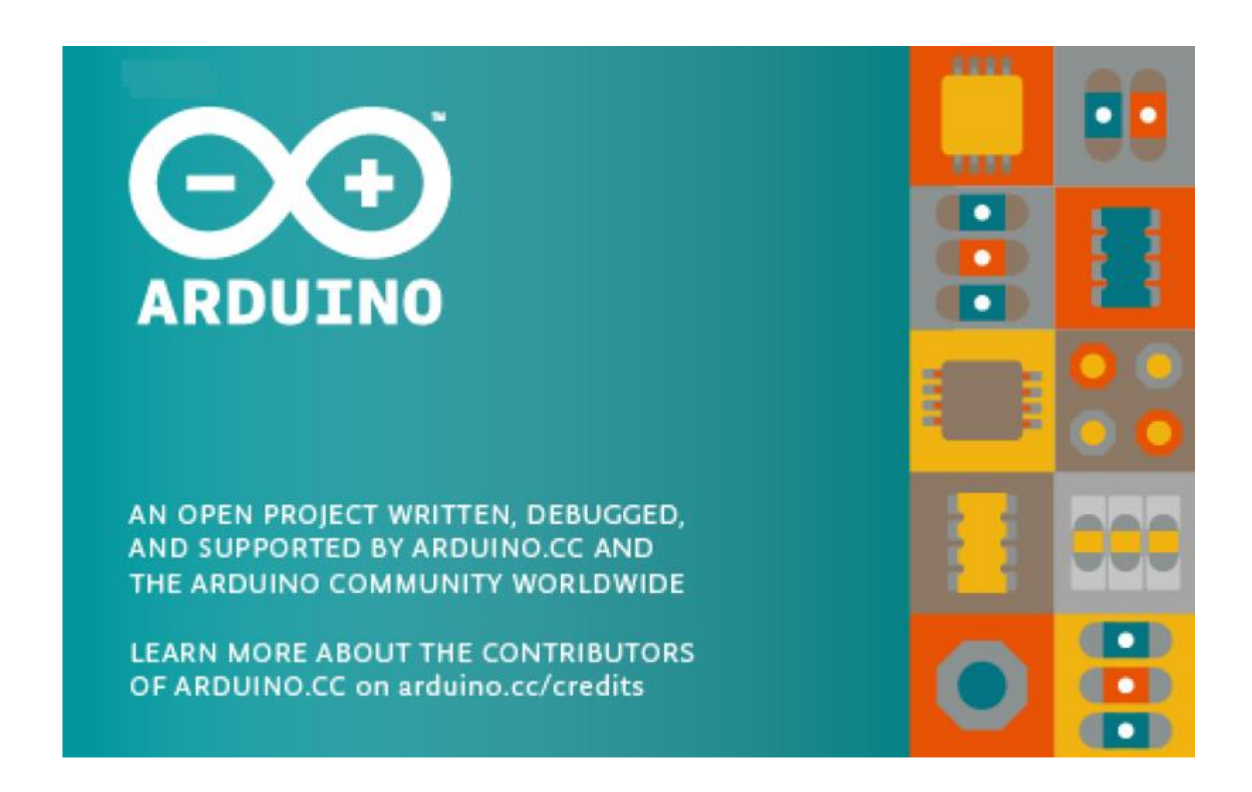
لذلك حتى نبدأ العمل في الاردوينو نحنا بحاجة الى ما يلي :

| اسم القطعة/الاداة | الشكل |
|---|---|
| Arduino Uno R3 | |
| كيبل USB<br><br>سوف نستخدمه لبرمجة الاردوينو وتغذيته وايضا استقبال المعلومات وعرضها على جهاز الحاسوب او شاشات العرض المختلفة | |
| لوحة توصيل Bread board<br><br>بحجم :<br>tie points 840 | |

| | |
|---|---|
| | اسلاك توصيل من نوع<br>Male-male<br>30 عدد |
| | اسلاك توصيل من نوع<br>female-male<br>30 عدد |
| | اسلاك توصيل من نوع<br>female- female<br>30 عدد |
| | 9V BATTERY<br>CONNECTOR FOR<br>ARDUINO |

| | |
|---|---|
| | UNO R3 CASE ENCLOSURE TRANSPARENT ACRYLIC BOX |
| | اضواء LEDمختلفة الالوان والاحجام |
| | PUSH BUTTON SWITCH عدد 10 |
| | عدة مقاومات بقيم مختلفة |

جميع القطع التي سبق ذكرناها حتى نبدأ في اساسيات التحكم والبرمجة باستخدام الاردوينو . لاحقا في هذا الكتاب وفي الجزء السادس سوف تزداد الادوات المطلوبة من حساسات ومحركات واغطية للاردوينو والكثير من الامور الممتعة .




Arduino IDE هي المكان الذي به نكتب الاكواد البرمجية باستخدام لغة الاردوينو (سي اردوينو ) يقوم البرنامج بالتعرف على الاردوينو وارسال الكودات الى اللوحة واستقبال وعرض المعلومات من اللوحة . فهي تتميز بسهولة الاستخدام ومجانية وتقوم بتحميل الكود مباشرة الى الاردوينو فباستخدام هذا البرنامج انت بغنى عن جميع البرامج الاخرى تقريبا.

هل يوجد برنامج غير هذه البيئة لبرمجة الاردوينو والتحكم به ؟

بالتأكيد ! , ان برمجة الاردوينو لها عدة طرق لبرمجتها لكن انت تختار الانسب حسب ما تراه مناسبا وممتعا لك , فمن هذه البرامج, برنامج الماتلاب وبرنامج اللاب فيو , ويوجد اكثر من طريقة للبرمجة يا اما باستخدام الكودات البرمجية او باستخدام البرمجة الصندوقية كما هو الحال في سيميلولينك – ماتلاب وبرنامج الاب فيو .

لكن في هذا الكتاب سوف نركز على بيئة IDE.

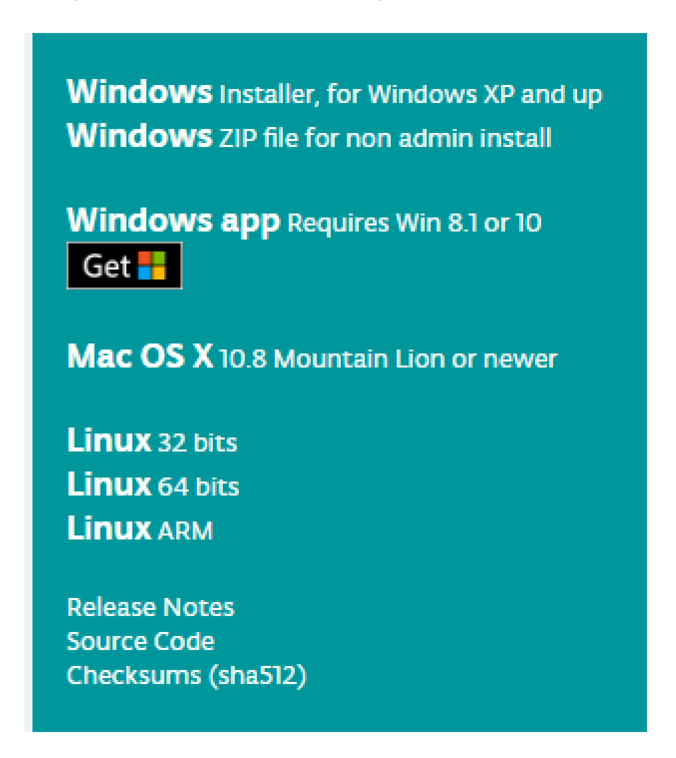
خطوات تنصيب بيئة الاردوينو IDE :

1 - في البداية سوف نقوم بتحميل البرنامج من الموقع الالكتروني الرسمي التالي , او من خلال القرص المدمج في الكتاب :




www.arduino.cc/en/main/software

2 - تختار نظام التشغيل الذي تعمل به (ماك , لينكس , ويندوز)

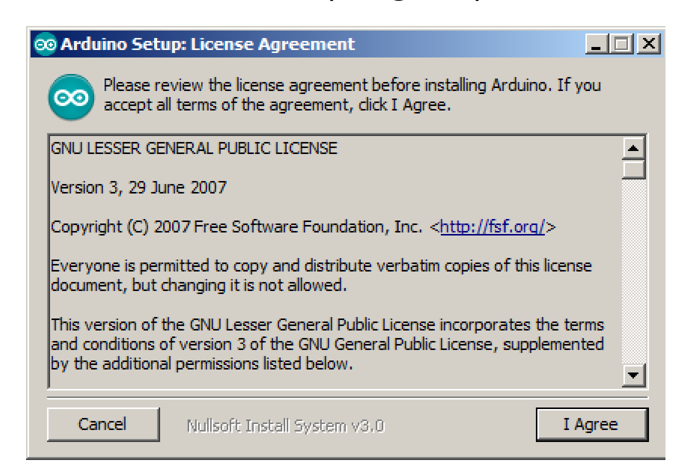


3 – سوف تظهر لك رسالة للمساهمة في دعم شركة اردوينو بالمبلغ الذي تراه مناسبا , لك الخيار في المساهمة او عدم المساهمة .

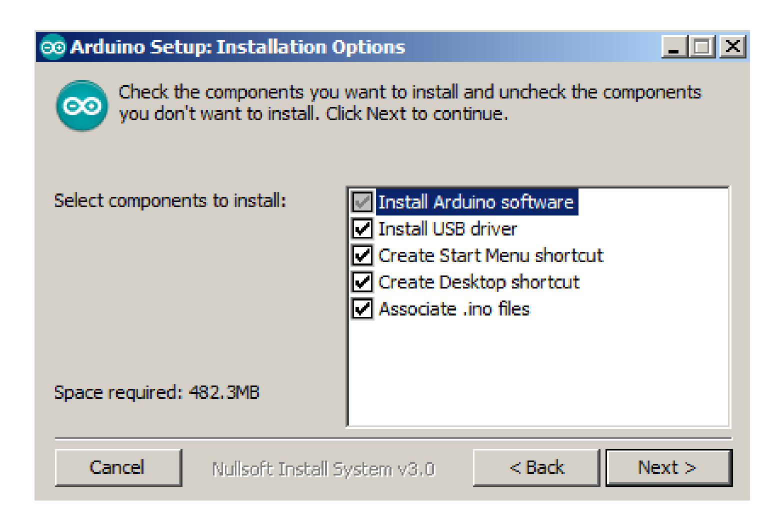



4 – انا استخدم نظام ويندوز 7 عند الضغط على ( Windows Installer, for Windows XP and up) سيبدأ تحميل البرنامج باصدار (1.8.8) في شهر يناير 2019

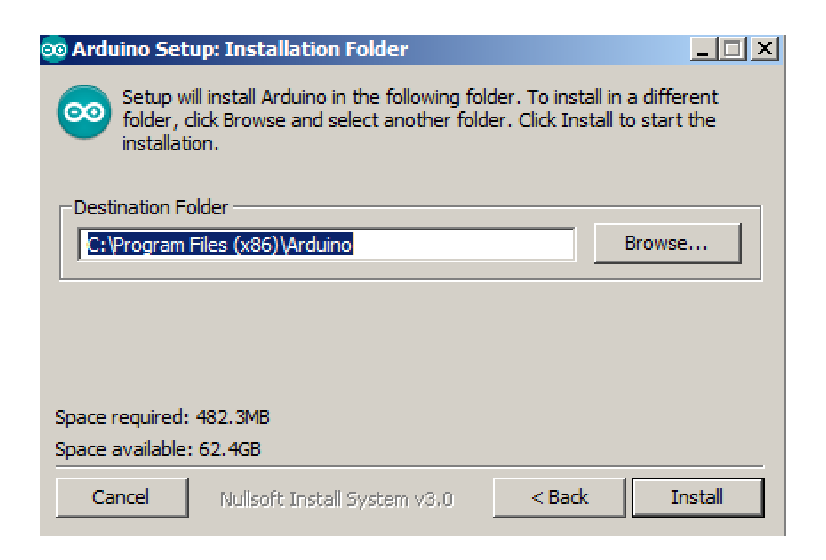
5 – قم بفتح ملف التحميل , لتنزيل البرنامج على جهاز الحاسوب وستظهر الرسالة التالية , قم بالضغط على موافق (I Agree) بعد قراءتك للتعليمات



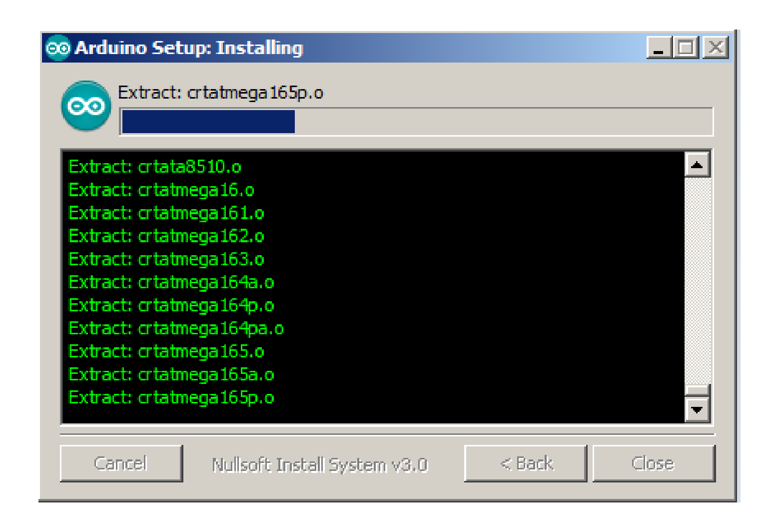
6 – تاكد من وضعك اشارة الصح داخل المربعات التالية



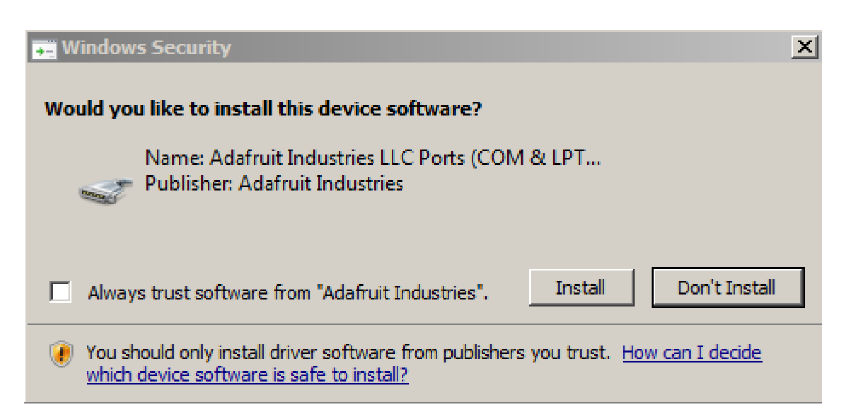

7 – اختر مكان تنزيل البرنامج



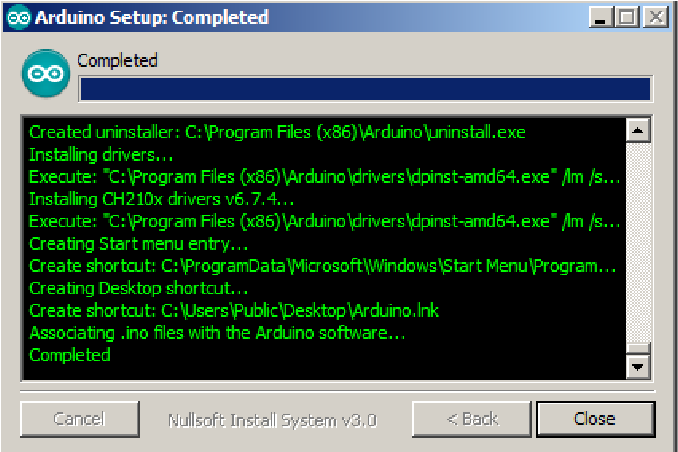
8 – سيبدأ البرنامج في التنصيب على جهاز الحاسوب




9 – عندما تظهر الرسالة التالية قم بالضغط على (Install) دائما




10 – عند اكتمال تنصيب البرنامج ستظهر النافذة التالية معنونة ب (Completed)



▪ الواجهة الاساسية للبرنامج

بعد تحميل البرنامج وفتح الشاشة الرئيسية تلاحظ البرنامج كالتالي :

sketch_jan17a | Arduino 1.8.8

File Edit Sketch Tools Help

sketch_jan17a

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

1 Arduino/Genuino Uno on COM1

لا بد انك لاحظت ان البرنامج صغير وهو خالي من الادوات والتعقيدات والكثير من التفاصيل فمن السهل فهم كيف يعمل للمبتدئ والمحترف .

مكونات البرنامج :

1 – شريط العنوان

sketch_jan17a | Arduino 1.8.8

يقوم البرنامج بتسمية افتراضية حسب التقويم

2 – شريط القوائم

File Edit Sketch Tools Help

يحتوي البرنامج على الادوات الهامة للتعامل مع المكتبات والكود ومنافذ ال USB اضافة الى تعريف انواع الاردوينو المختلفة .

| اللائحة | ابرز استخداماتها |
|---|---|
| File | اعدادات البرنامج الاساسية<br>حفظ وفتح الكودات<br>استعراض الامثلة التي يوفرها البرنامج |
| Edit | النسخ واللصق<br>الانتقال والبحث في الاسطر |
| Sketch | التاكد من صحة الكود<br>تحميل الكود الى الاردوينو<br>اضافة المكتبات والملفات |
| Tools | تعريف المنفذ التسلسلي<br>تعريف نوع الاردوينو<br>تعريف المعالج المستخدم في الاردوينو<br>واخذ معلومات الاردوينو |
| Help | دليل استخدام البرنامج والاردوينو |

3 – شريط الاوامر السريعة




4 – حيز العمل

```
void setup() {
  // put your setup code here, to run once:

}

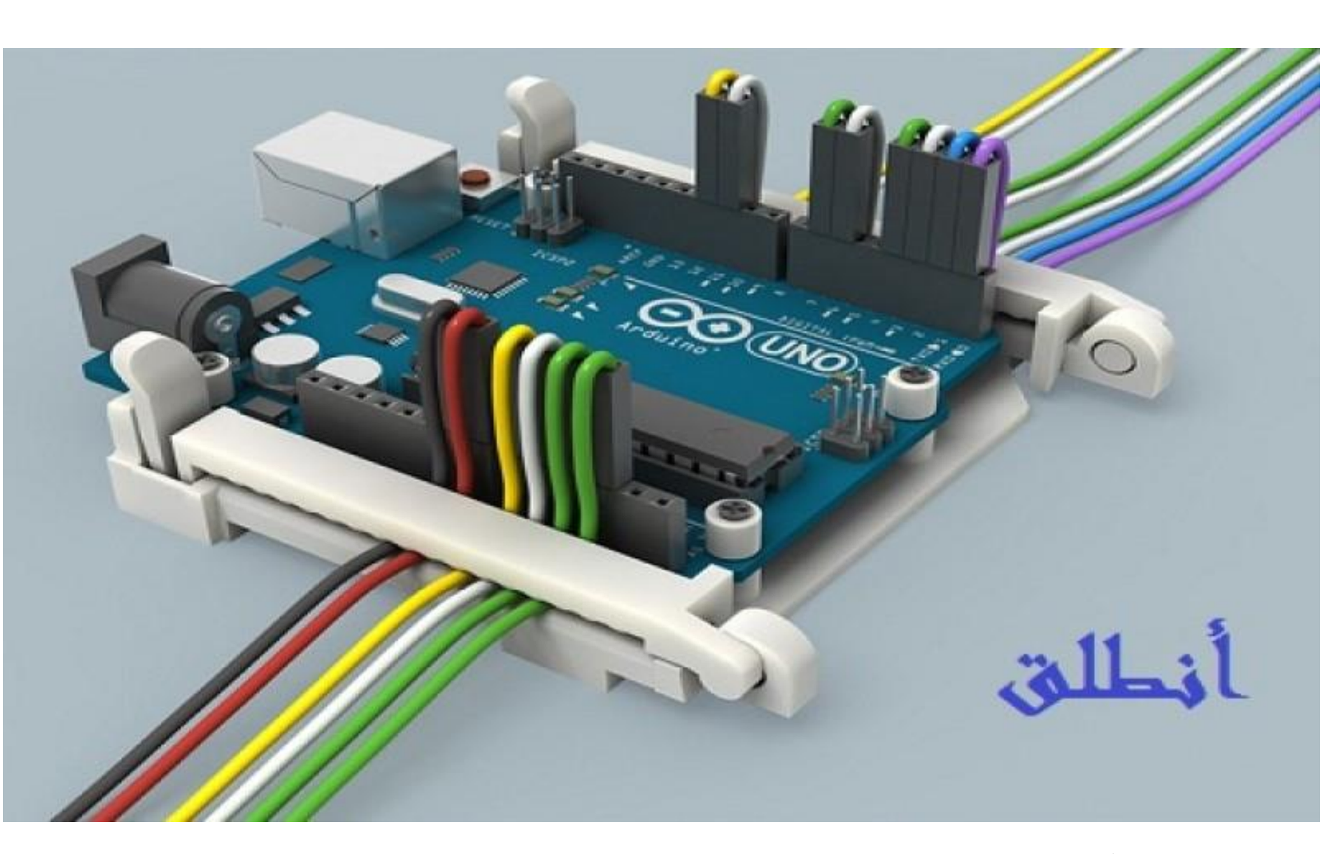
void loop() {
  // put your main code here, to run repeatedly:

}
```

5 – شريط التنبيهات

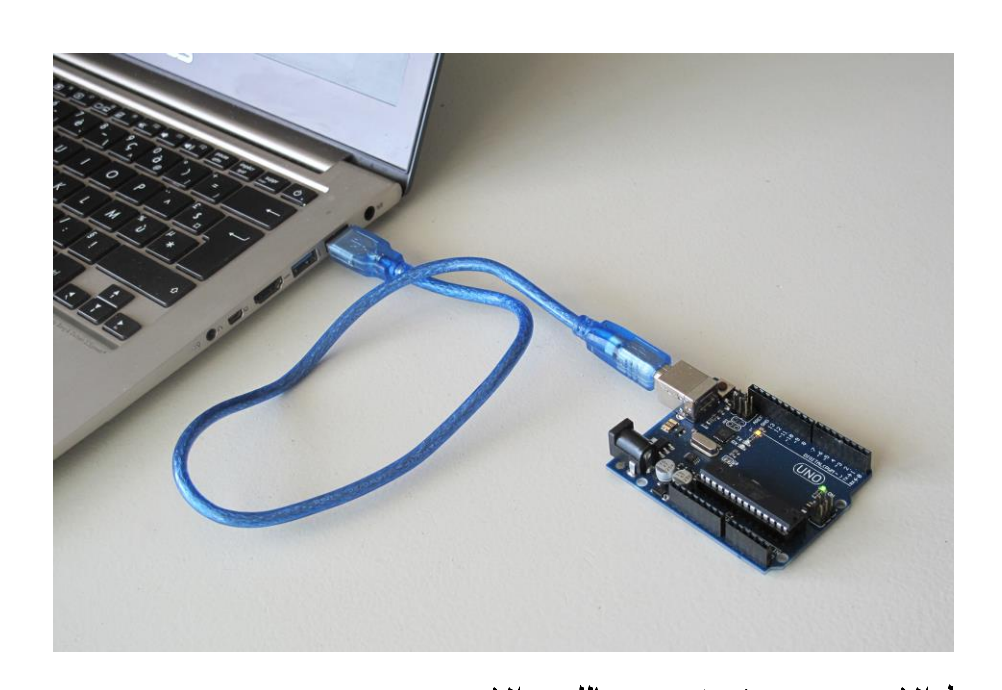


# تهيئة الاردوينو



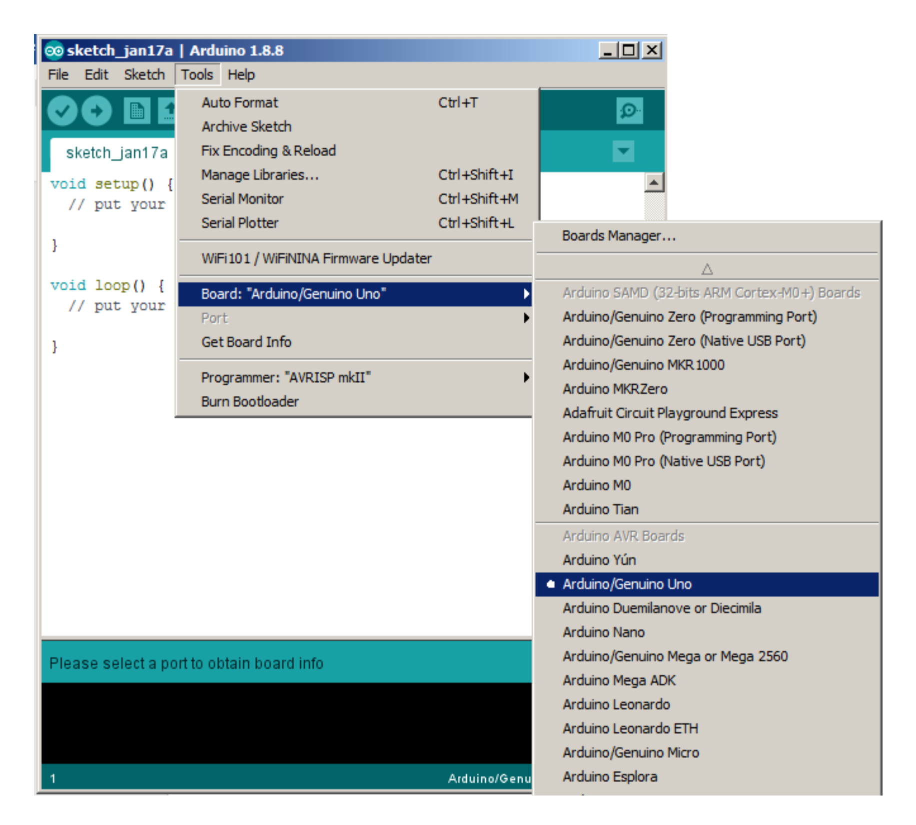

قبل الانطلاق في برمجة الاردوينو يجب تعريفه على جهاز الحاسوب كما يلي :

ربط الاردوينو مع جهاز الحاسوب باستخدام ال USB



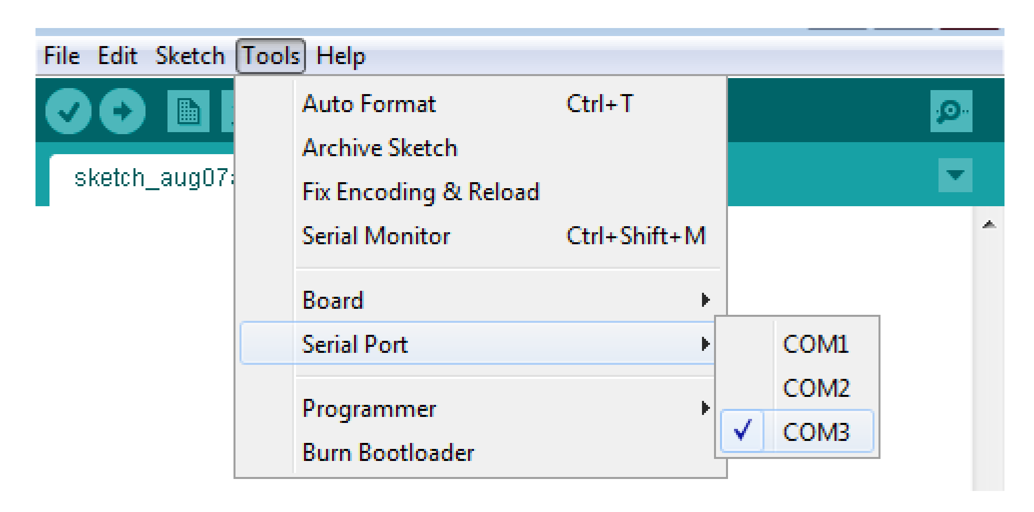

عند ربط الاردوينو سوف تضيئ باللون الاخضر

نفتح البرنامج وتختار نوع الاردوينو الذي تستخدمه , انا ساستخدم النوع اونو , من
Tools – Board




ويجب اختيار المنفذ الي به الاردوينو موصول

Tools - Port




الان بعد تعريف منفذ الاردوينو ونوع الاردوينو المستخدم يمكنك البدء في كتابة اول كود لتحميلة على الاردوينو مباشرة !

في نافذة العمل ستلاحظ الكود الاساسي التالي :

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

يتكون الكود من جزئين :

- الجزء الاول : قسم تهيئة المدخلات والمخرجات

```
void setup() {
  // put your setup code here, to run once:

}
```

- الجزء الثاني : قسم الكود الرئيسي الذي سيعمل به الاردوينو

```
void loop() {
  // put your main code here, to run repeatedly:

}
```

- انواع المتغيرات الاساسية في C Arduino:

| النوع | القيمة | مثال |
|---|---|---|
| integer | -32768 الى 32768 | int motor = 5 ;<br>int led_1 = 13; |
| float | الاعداد العشرية | float data = 5.8; |
| long | -2,147,483,648 الى<br>2,147,483,648 | long = 12111221; |
| byte | 0 الى 225 | byte x = 125; |
| character | حروف نصية او كلمات | char name = ardu; |

- الكلمات المحجوزة المشهورة مرتبة في الجدول التالي , ولمشاهدة جميع الكلمات انظر الملحق A-2 في نهاية الكتاب

| | | | |
|---|---|---|---|
| HIGH | pinMode() | analogRead() | for |
| LOW | digitalWrite() | analogReference() | while |
| if | digitalRead() | analogWrite() | do while |
| if - else | delay | - | Serial.begin() |
| switch | - | - | Serial.println |

- اولًا كلمات معالجة الاشارات الرقمية :

1 - pinMode()

تستخدم للتحكم في الاطراف الرقمية لجعلها كمدخلات أو مخرجات , والصيغة العامة لها كالتالي :

```
pinMode(pin, mode)
```

حيث ان :

| | |
|---|---|
| pinMode | الدالة/الاقتران/الامر |
| pin | رقم الطرف الرقمي |
| mode | مدخل او مخرج<br>INPUT Or OUTPUT |

مثلا لجعل الطرف رقم 13 مخرج نقوم بعمل التالي :

```
void setup()
{
  pinMode(13, OUTPUT);     // sets the digital pin 13 as output
}
```

لنجعل الطرف رقم 8 مدخل نقوم بعمل التالي :

```
void setup()
{
  pinMode(13, INPUT);     // sets the digital pin 13 as output
}
```

## 2 - digitalWrite()

تستخدم لتحميل اوامر التشغيل والايقاف الى اطراف الاردوينو الرقمية , فاذا كان لديك اردوينو اونو يقوم هذا الامر باخراج اشارة قيمتها 5 فولت او 0 فولت حسب كود المستخدم, والصيغة العامة لها كالتالي :

```
digitalWrite(pin, value)
```

حيث ان

| الدالة/الاقتران/الامر | pinMode |
|---|---|
| رقم مخرج الطرف الرقمي | pin |
| تشغيل او ايقاف<br>HIGH Or LOW | Value |

مثلا لجعل الطرف رقم 13 مخرج للتشغيل نعمل التالي :

```
digitalWrite(13, HIGH);
```

لجعل الطرف رقم 13 مخرج للايقاف نعمل التالي :

```
digitalWrite(13, LOW);
```

## 3 - digitalRead()

يقرأ القيمة من دبوس رقمي محدد ، إما HIGH أوLOW , والصيغة العامة لها كالتالي :

```
digitalWrite(pin)
```

حتى نستطع استخدام هذا الامر يجب في البداية تعين متغير ثم استخدام PinMode لتحديد المدخلات ثم نقوم بتخزين قيمة الطرف داخل متغير معرف مسبقا كالتالي:

```
int inPin = 7;
int val = 0;        // variable to store the read value
void setup()
{
```

```
  pinMode(inPin, INPUT);  // sets the digital pin 7 as input
}
void loop()
{
  val = digitalRead(inPin);     // read the input pin

}
```

- ثانيا كلمة التأخير الزمني الدقيق **delay ()**

تستخدم لايقاف البرنامج مؤقتًا لمقدار من الوقت (بالميلي ثانية) مثلا عند ادخال 1000 هي تعني 1 ثانية

مثلا برنامج لتشغيل الطرف 13 مدة 3 ثواني :

```
  digitalWrite(ledPin, HIGH);   // sets the LED on
  delay(3000);
```

▪ ثالثا الكلمات الشرطية :

تحتوي لغة C Arduino على العديد من الكلمات الشرطية مثل الكلمات التي درسناها في الجزء الثالث من السلسلة في كتاب مقدمة في البرمجة عندما استخدمنا ++C , ومن هذه الكلمات ما يلي :

| | |
|---|---|
| if | if-else |
| switch | - |

استخدام الجملة if :

يتم تنفيذ الشرط فقط اذا تحقق الشرط , واذا لم يتحقق الشرط يكمل البرنامج عمله والصيغة العامة لها هي :

```
if (condition)
{
  //statement(s)
}
```

يمكن ان تكون الاوامر statement عبارة عن عمليات مقارنة مثل :

```
x == y (x is equal to y)
x != y (x is not equal to y)
x <  y (x is less than y)
x >  y (x is greater than y)
x <= y (x is less than or equal to y)
x >= y (x is greater than or equal to y)
```

مثال لطرق تمثيل الشرط باستخدام الجملة if

```
if (x > 120) digitalWrite(LEDpin, HIGH);


or
```

```
if (x > 120)
digitalWrite(LEDpin, HIGH);

or
if (x > 120){ digitalWrite(LEDpin, HIGH); }

if (x > 120){
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}
```

استخدام الجملة if-else :

تمثيل  آخر للشرط يتيح تحكمًا أكبر في البرنامج أكثر من الشرط if الأساسي ، عن طريق السماح بتجميع عدة اختبارات معًا. سيتم تنفيذ جملة أخرى (إذا وجدت) إذا كان الشرط الموجود في العبارة if ينتج عنه false. يمكن أن يشرع الآخر في اختبار آخر بحيث يمكن تشغيل العديد من الاختبارات الحصرية للطرفين في نفس الوقت, والصيغة العامة لها هي :

```
if (condition1)
{
  // do Thing A
}
else if (condition2)
{
  // do Thing B
}
else
{
  // do Thing C
}
```

مثال :

```
if (temperature >= 70)
{
  //Danger! Shut down the system
}
else if (temperature >= 60 && temperature < 70)
{
  //Warning! User attention required
}
else
{
  //Safe! Continue usual tasks...
}
```

استخدام الجملة switch :

يتحكم اختيار جواب الشرط خلال السماح للمبرمجين بتحديد رمز مختلف يجب تنفيذه في ظروف مختلفة. على وجه الخصوص ، يقارن عبارة switch قيمة المتغير بالقيم المحددة في عبارات الشرط, وعند العثور على بيان حالة تتطابق قيمته مع متغير المتغير ، يتم تشغيل التعليمة البرمجية في هذه الحالة وتستخدم الكلمة break عادةً في نهاية كل حالة , والصيغة العامة لها هي :

```
switch (var) {
  case 1:
    // statements
    break;
  case 2:
    // statements
    break;
  default:
    // statements
    break;
}
```

حيث ان :

| switch | الدالة/الاقتران/الامر |
|---|---|
| (var) | المتغير الذي تقارن قيمته مع الحالات المختلفة ويقبل فقط متغيرات من نوع int, char |
| 2 - 1 | اسماء ثوابت |

مثال :

```
switch (range) {
    case 0:    // your hand is on the sensor
      Serial.println("dark");
      break;
    case 1:    // your hand is close to the sensor
      Serial.println("dim");
      break;
    case 2:    // your hand is a few inches from the sensor
      Serial.println("medium");
      break;
    case 3:    // your hand is nowhere near the sensor
      Serial.println("bright");
      break;
  }
```

- رابعا كلمات معالجة الاشارات التماثلية :

## 1 - analogRead()

يقوم بقراءة القيمة التناظرية من الدبوس التناظري المحدد. تحتوي لوحات اردوينو على محول متعدد القنوات تناظري إلى رقمي 10-بت. هذا يعني أنه سوف يقوم بتخطيط جهد المدخلات بين 0 و جهد التشغيل (5 V أو 3.3 V) إلى قيم صحيحة بين 0 و 1023. على Arduino UNO ، على سبيل المثال ، ينتج عن هذا حل بين قراءات: 5 فولت / 1024 وحدة أو ، 0.0049 فولت (4.9 فولت) لكل وحدة. انظر الجدول أدناه للحصول على دبابيس قابلة للاستخدام ، جهد التشغيل والحد الأقصى لبعض لوحات اردوينو.يمكن تغيير نطاق الإدخال باستخدام analogReference () .

على لوحات ATmega القائمة ( UNO، Nano، Mini، Mega) ، يستغرق حوالي 100 ميكروثانية (0.0001 ثانية) لقراءة مدخلات تناظرية ، وبالتالي فإن الحد الأقصى لمعدل القراءة هو حوالي 10000 مرة في الثانية.

| اللوحة | جهد التشغيل | عدد الدبابيس المتوفرة | اعلى قراءة تناظرية |
|---|---|---|---|
| Uno | 5 Volts | A0 to A5 | 10 bits |
| Mini, Nano | 5 Volts | A0 to A7 | 10 bits |
| Mega, Mega2560, MegaADK | 5 Volts | A0 to A14 | 10 bits |
| Micro | 5 Volts | A0 to A11 | 10 bits |
| Leonardo | 5 Volts | A0 to A11 | 10 bits |
| Zero | 3.3 Volts | A0 to A5 | 12 bits** |
| Due | 3.3 Volts | A0 to A11 | 12 bits** |
| MKR Family boards | 3.3 Volts | A0 to A6 | 12 bits** |

** تعني دقة analogRead () الافتراضية لهذه اللوحات هي 10 بت ، للتغير يجب استخدام analogReadResolution () لتغييره إلى 12 بت.

الصيغة العامة لهذه الدالة هي :

```
analogRead(pin)
```

pin اسم دبوس الإدخال التمثلي للقراءة من A0 إلى A5 على معظم اللوحات ، A0إلى A6 على لوحات MKR ، A0 إلى A7 على Mini و Nano ، A0إلى A15على Mega

قيمة القراءة التناظرية على الدبوس بالمتغير (int). على الرغم من أنه يقتصر على دقة المحول الرقمي إلى الرقمي (0-1023 لـ 10 بتات أو 0-4095 لـ 12 بت).

سوف نكتب برنامج يقرأ قيمة الجهد على دبوس تناظري بحيث تتغير باستخدام مجزء الجهد او مقاومة متغيرة ويعرضها للمستخدم باستخدام دالة جديدة اسمها **Serial.println ()** و **Serial.begin()**

```
int analogPin = A3; /* potentiometer wiper (middle terminal)
*/connected to analog pin 3
                       // outside leads to ground and +5V
int val = 0;           // variable to store the value read

void setup()
{
  Serial.begin(9600);            //  setup serial
}

void loop()
{
  val = analogRead(analogPin);   // read the input pin
  Serial.println(val);           // debug value
}
```

## 2 - analogReference()

تستخدم لتهيئة الجهد المرجعي المستخدم في الإدخال التناظري (أي القيمة المستخدمة كأعلى نطاق للإدخال). الخيارات التي يوفرها الاردوينو حسب النوع هي:

Arduino AVR Boards (Uno, Mega, etc.)

- DEFAULT: the default analog reference of 5 volts (on 5V Arduino boards) or 3.3 volts (on 3.3V Arduino boards)
- INTERNAL: an built-in reference, equal to 1.1 volts on the ATmega168 or ATmega328P and 2.56 volts on the ATmega8 (not available on the Arduino Mega)
- INTERNAL1V1: a built-in 1.1V reference (Arduino Mega only)
- INTERNAL2V56: a built-in 2.56V reference (Arduino Mega only)
- EXTERNAL: the voltage applied to the AREF pin (0 to 5V only) is used as the reference.

Arduino SAMD Boards (Zero, etc.)

- AR_DEFAULT: the default analog reference of 3.3V
- AR_INTERNAL: a built-in 2.23V reference
- AR_INTERNAL1V0: a built-in 1.0V reference
- AR_INTERNAL1V65: a built-in 1.65V reference
- AR_INTERNAL2V23: a built-in 2.23V reference
- AR_EXTERNAL: the voltage applied to the AREF pin is used as the reference

Arduino SAM Boards (Due)

- AR_DEFAULT: the default analog reference of 3.3V. This is the only supported option for the Due.

الصيغة العامة لهذه الدالة هي :

```
analogReference(type)
```

(type): أي نوع من الخيارات للاستخدام (انظر قائمة الخيارات في الوصف اعلاه )

بعد تغيير المرجع التناظري ، قد لا تكون القراءات القليلة الأولى من analogRead ()دقيقة.

لا تستخدم أي شيء أقل من 0 فولت أو أكثر من 5 فولت للجهد المرجعي الخارجي على دبوس AREF إذا كنت تستخدم مرجعًا خارجيًا على دبوس AREF ، فيجب تعيين الإشارة التناظرية إلى EXTERNAL قبل استدعاء analogRead (). خلاف ذلك ، سوف تقوم باختصار الجهد المرجعي النشط (الذي يتم إنشاؤه داخليًا) و دبوس AREF ، مما قد يؤدي إلى تلف وحدة التحكم الدقيقة في لوحة Arduino.

بدلاً من ذلك ، يمكنك توصيل جهد المرجع الخارجي إلى دبوس AREF من خلال المقاومة K 5، مما يسمح لك بالتبديل بين الفولطية المرجعية الخارجية والداخلية. لاحظ أن المقاوم سيغير الجهد الذي يتم استخدامه كمرجع لأن هناك المقاوم K 32الداخلي على دبوس AREF. وهما بمثابة مقسم الجهد ، لذلك ، على سبيل المثال ، V 2.5 المطبق من خلال المقاوم ستنتج 2.5 * 32 / (32 + 5) = ~ 2.2 Vفي دبوس AREF.

## 3 - analogWrite()

تستخدم لكتابة (تحميل) القيمة التماثلية (موجة PWM) الى الدبوس , فمثلا يمكن استخدامه للتحكم في اضاءة مصباح LED بدرجات متفاوتة او قيادة محرك بسرعات مختلفة كما سندرس هذه الامثلة في هذا الكتاب , بعد استدعاء analogWrite() سيقوم الدبوس بتوليد موجة مربعة ثابتة لحتى يتم استدعاء analogWrite() او digitalRead() او digitalWrite() على نفس الدبوس , إن تردد إشارة PWM على معظم الدبابيس يبلغ حوالي 490 هرتز , بينما على لوحات Uno واللوحات المماثلة ، يكون للدبوسين 5 و 6 تردد حوالي 980 Hz.

الصيغة العامة لهذه الدالة هي :

```
analogWrite(pin, value)
```

حيث يسمح استخدام النوع int فقط , ودورة التشغيل: بين 0 و 255 (دائمًا).

مثال للتحكم في شدة اضاءة مصباح LED

```
int ledPin = 9;       // LED connected to digital pin 9
int analogPin = 3;    // potentiometer connected to analog
pin 3
int val = 0;          // variable to store the read value

void setup()
{
  pinMode(ledPin, OUTPUT);   // sets the pin as output
}

void loop()
{
  val = analogRead(analogPin);   // read the input pin
  analogWrite(ledPin, val / 4);  // analogRead values go
from 0 to 1023, analogWrite values from 0 to 255
}
```

▪ خامسا الدوران :

تحتوي لغة Arduino C على العديد من الكلمات الدورانية مثل الكلمات التي درسناها في الجزء الثالث من السلسلة في كتاب مقدمة في البرمجة عندما استخدمنا C++ , ومن هذه الكلمات ما يلي :

| | |
|---|---|
| while | for |
| - | do while |

استخدام الجملة for:

يتم استخدام العبارة for لتكرار كتلة من العبارات حيث يتم استخدام عداد الزيادة عادةً لزيادة و إنهاء الحلقة. تعتبر عبارة for مفيدة لأي عملية متكررة و هي اكثر الحلقات الدورانية مرونة ، وغالبًا ما يتم استخدامها مع المصفوفات للعمل على مجموعات من البيانات / دبابيس.

الصيغة العامة لهذه الدالة هي :

```
for (initialization; condition; increment) {
     //statement(s);
}
```

| | |
|---|---|
| بداية الحلقة الدورانية | initialization |
| اختبار الشرط ؛ إذا كان صحيحا او لا | condition |
| يتم تنفيذ الزيادة او النقصان او الامر الموجود ، ثم يتم اختبار الشرط مرة أخرى. عندما يصبح الشرط false ، تنتهي الحلقة. | increment |

مثال على جملة for

```
// Dim an LED using a PWM pin
```

```
int PWMpin = 10; // LED in series with 470 ohm resistor on
pin 10

void setup()
{
  // no setup needed
}

void loop()
{
   for (int i=0; i <= 255; i++){
      analogWrite(PWMpin, i);
      delay(10);
   }
}
```

مثال آخر ، يتلاشى LED صعودا وهبوطا باستخدام الحلقة for :

```
void loop()
{
   int x = 1;
   for (int i = 0; i > -1; i = i + x){
      analogWrite(PWMpin, i);
      if (i == 255) x = -1;             // switch
direction at peak
      delay(10);
   }
}
```

استخدام الجملة الدورانية while:

الصيغة العامة لهذه الدالة هي :

```
while(condition){
  // statement(s)
}
```

الشرط عبارة عن تعبير منطقي يتم تقييمه إلى true أوfalse

مثال

```
var = 0;
while(var < 200){
  // do something repetitive 200 times
  var++;
}
```

استخدام الجملة الدورانية do while

تعمل بنفس الطريقة مثل حلقة while ، باستثناء أنه يتم اختبار الشرط في نهاية الحلقة ، بحيث يتم تشغيل الحلقة دائمًا مرة واحدة على الأقل.

الصيغة العامة لهذه الدالة هي :

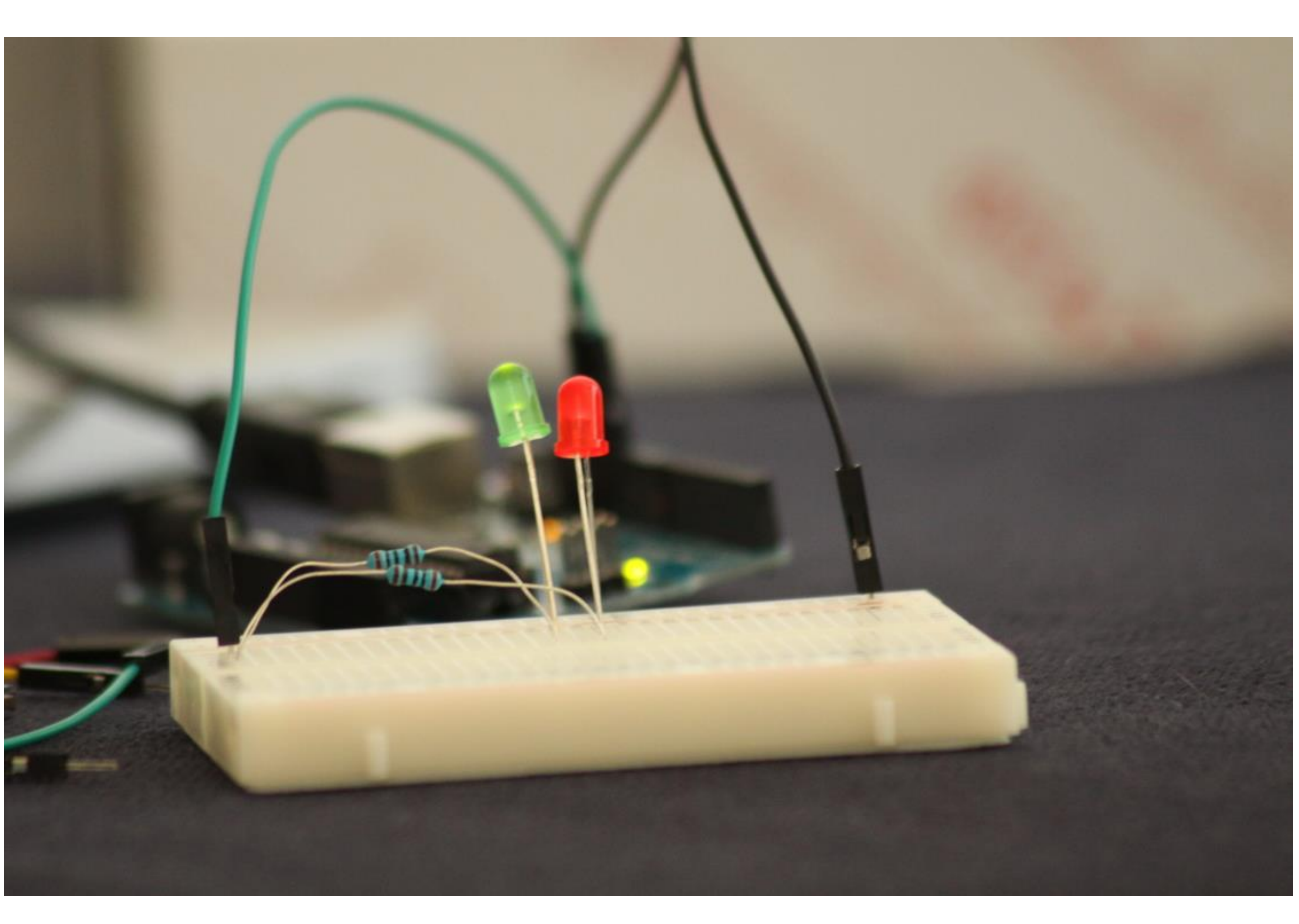
```
do
{
    // statement block
} while (condition);
```

مثال

```
int x = 0;
do
{
  delay(50);           // wait for sensors to stabilize
  x = readSensors();   // check the sensors

} while (x < 100);
```

# المصابيح LED



في هذا الدرس سوف نقوم بكتابة برامج للتحكم في تشغيل واطفاء المصابيح الصغيرة LED حيث نوضح كيفية ربط المصباح بالمفاتيح وبرمجتها على فترة تاخير زمنية معينة والمزيد من الامثلة الممتعة .

- المثال الاول : تشغيل الدايود الضوئي LED بفترة زمنية

يتكون البرنامج الأول الذي يتعلمه كل مبرمج من كتابة الكود ما يكفي من الكود لإظهار الرمز الخاص به في جملة "Hello World!" على الشاشة. اما الاردوينو تشغيل الاضواء هو "Hello World!" اذا اردناها بصيغة الحوسبة الفيزيائية.

| القطع المطلوبة | |
|---|---|
| LED | ohm resistor 220 |
| Arduino uno | Wires |
| Power tool (اختياري) | Bread board(اختياري) |

مصباح LED هو مصباح صغير (وهو يقف في اتجاه "الصمام الثنائي الباعث للضوء") الذي يعمل بجهد قليل نسبيًا. يحتوي لوح Arduino على طرف رقمي رقم 13 وهو الذي سوف نستخدمه في البرمجة لهذا المثال

مصباح LED يستغرق سوى عدة أسطر فقط من البرمجة. أول شيء نقوم به هو تحديد متغير سيحمل رقم الطرف التي يتصل بها .يتعين علينا القيام بذلك بحيث نستخدم متغير صحيح يسمى(int ) الذي سبق درسناه في لغة C++

```
int ledPin = 13;
```

الشيء الثاني الذي يجب علينا القيام به هو تكوين مخرج متصل بالـ LED نحن نفعل هذا باستخدام جملة pinMode() داخل void setup() كما يلي :

```
void setup()
{
  pinMode(ledPin, OUTPUT);
}
```

في الخطويتين السابقتين , قمنا بتعريف طرف الاردوينو الذي نعمل به وتعينه كمخرج الآن يجب عليها ادخال الاوامر في الجزء الثاني وهو void loop()

- الامر digitalWrite ()

يستخدم هنا للتحكم في التشغيل والايقاف , عند التشغيل نستخدم الكلمة HIGH وعند الايقاف نستخدم الكلمة LOW

- الامر delay ()

يستخدم هذا الامر لاضافة الفترة الزمنية , يجب العلم ان الاردوينو من الانظمة ذات الزمن الحقيقي فهو دقيق يتعامل مع اجزاء الثانية , فاذا اردنا اضاءة المصباح ثانية واحدة سوف نكتب 1000

تكون صيغة اوامر البرنامج كالتالي

```
void loop()
{
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```

الآن لقد انتهينا من كتابة الكود في ثلاثة خطوات , ملخص ما قمنا به :

1 – تعريف المتغير ورقم الطرف باستخدام int

2 – تكوين المخرج باستخدام pinMode

3 – التشغيل والايقاف بفترة زمنية باستخدام digitalWrite و delay

بعد تجميع الكود ووضعه في البرنامج سيكون في الصورة النهائية التالية :

sketch_jan17a §

```
int ledPin = 13;                 // LED connected to digital pin 13

void setup()
{
  pinMode(ledPin, OUTPUT);       // sets the digital pin as output
}

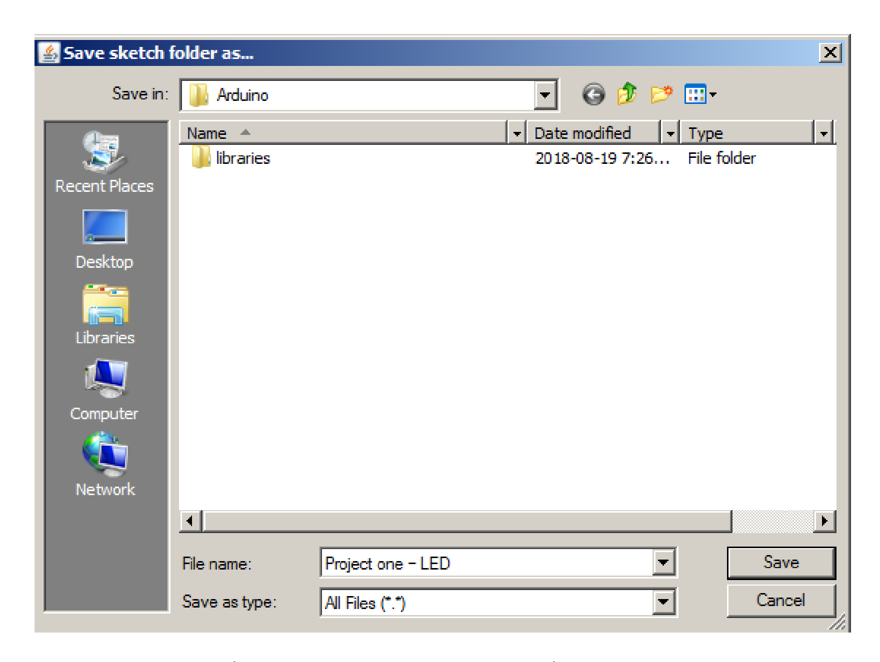
void loop()
{
  digitalWrite(ledPin, HIGH);    // sets the LED on
  delay(1000);                   // waits for a second
  digitalWrite(ledPin, LOW);     // sets the LED off
  delay(1000);                   // waits for a second
}
```

تذكر استخدمنا // لوضع الملاحظات داخل الكود كما تعلمنا في C++

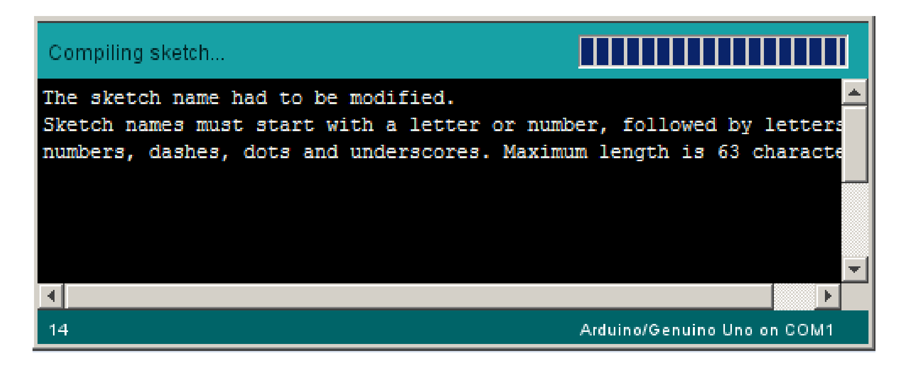
استخدامات صندوق الادوات السريعة ما يلي :

| الوظيفة | الشكل |
|---|---|
| Verify<br>للتاكد من صحة الكود | |
| Upolad<br>لتحميل الكود الى الاردوينو | |
| New<br>لانشاء ملف جديد | |
| Open<br>لفتح ملف سابق | |
| Save<br>لحفظ العمل | |
| Serial Monitor<br>نافذة عرض بيانات الاردوينو | |

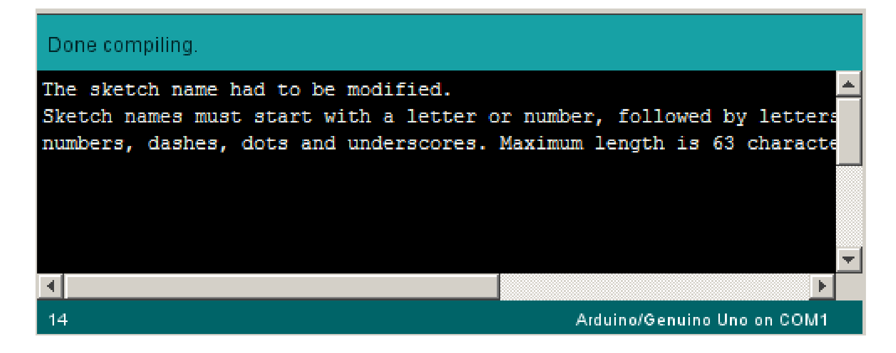
الآن سوف نضغط على زر التاكد من صحة الكود (verify) سوف نلاحظ انه يطلب منك حفظ الملف باسم معين , فلنسميه مثلا Project one – LED




وعند الضغط على زر الحفظ سيبدأ شريط الحالة بعملية التأكد من صحة الكود



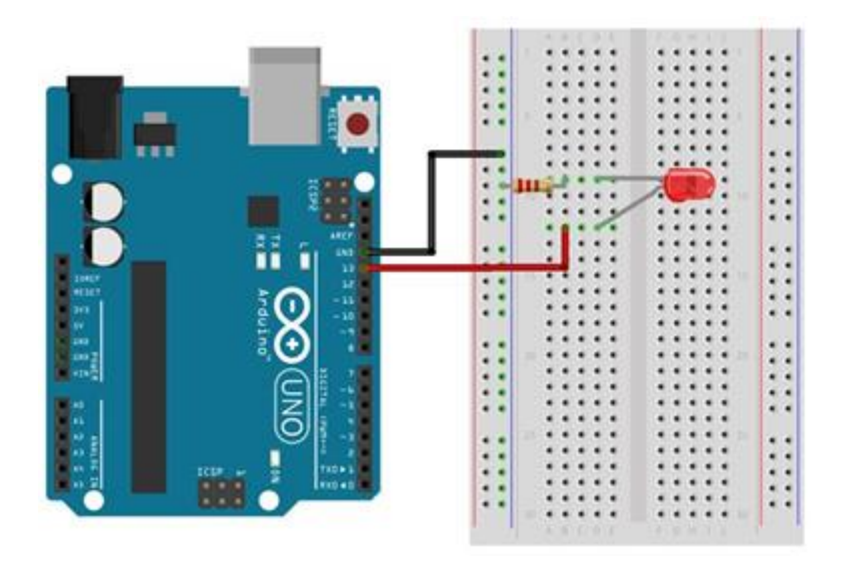

وعند الانتهاء تظهر الرسالة التالية (Done compiling)



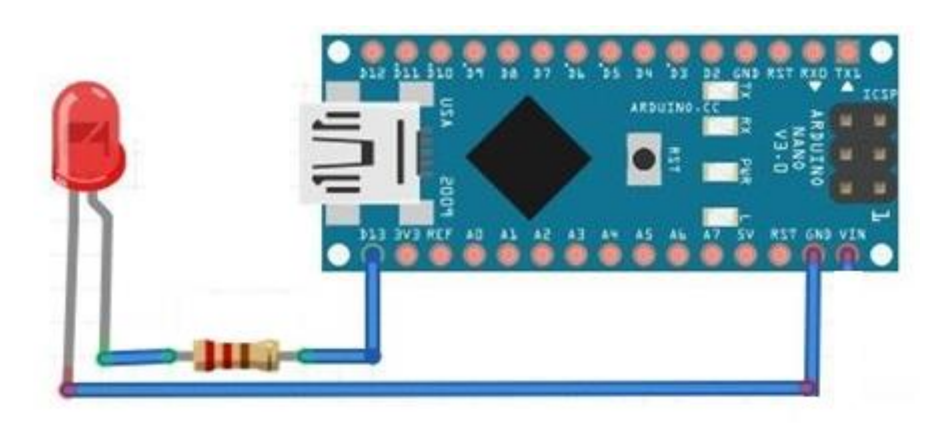

اذا لم تظهر الرسالة (Done compiling) سقوم البرنامج بتنبيهك بوجود خطأ.

طريقة توصيل القطع ستكون كالتالي :

باستخدام اردوينو اونو




باستخدام اردوينو نانو

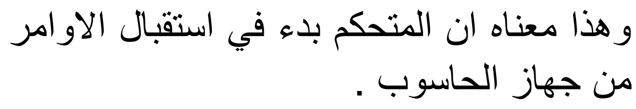



نلاحظ ان جميع انواع الاردوينو تعمل في نفس كود الاردوينو اذا تم استخدام الكود وطريقة التوصيل الصحيحة .

الآن بعد توصيلة الاردوينو كتابة الكود نحمل الكود على الاردوينو بالضغط على Upload



سنلاحظ خلال ثوان قليلة ان اضوية الاردوينو بدئت تضوي التي بجانبها (TX-RX) وهذا معناه ان المتحكم بدء في استقبال الاوامر من جهاز الحاسوب .



وسيدأ عمل الكود كما شرحناه سابقا , سوف يضيئ المصباح ثانية وينطفئ ثانية اخرى بشكل متكرر .

الآن سنغير مخرج الاردوينو من 13 الى 8 وتغير الوقت من ثانية الى 3 ثوان في التشغيل و 2 ثانية في الايقاف فصيبح الكود كالتالي :

```
int ledPin = 8;                 // LED connected to digital pin 8

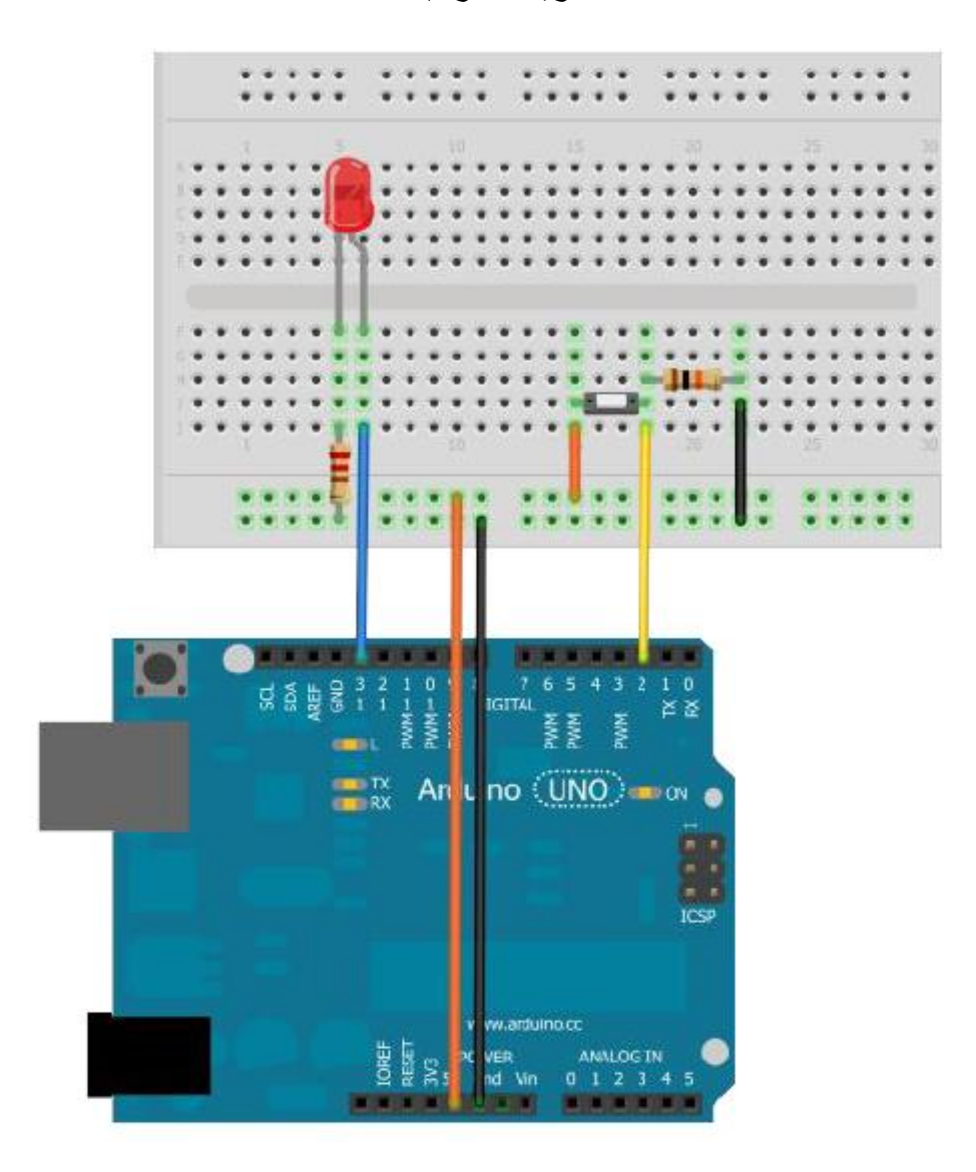
void setup()
{
  pinMode(ledPin, OUTPUT);      // sets the digital pin as output
}

void loop()
{
  digitalWrite(ledPin, HIGH);   // sets the LED on
  delay(3000);                  // waits for 3 seconds
  digitalWrite(ledPin, LOW);    // sets the LED off
  delay(2000);                  // waits for 2 seconds
}
```

- المثال الثاني : تشغيل الدايود الضوئي LED باستخدام مفتاح Button

| القطع المطلوبة | |
|---|---|
| 560 ohm resistor | LED |
| 10 kohm resistor | Push button |
| Bread board | Arduino uno |
| Wires | Power tool (اختياري) |

طريقة التركيب



الوصف :

في هذا المثال سنقوم ببرمجة تشغيل المصباح عند الضغط على المفتاح فقط .

خطوات العمل :

قم بوضع المفتاح على لوحة التوصيل ثم وصل احدى طرفيه بمدخل الطاقة 5V باستخدام الاسلاك والطرف الاخر بالمقاومة ذات القيمة 10 كيلو اوم مع طرف المدخل رقم 2 في الاردوينو .

تعمل هذه الدائرة على توفير دخل رقمي , فعند الضغط على الزر تنطلق نبضة بقيمة 5 فولت والتي تعتبرها الاردوينو اشارة من النوع HIGH وعند ترك الزر يفصل التيار الكهربائي ويصبح الدخل صفر فولت والتي تمثل LOW

```
int ledpin = 13;
int buttonPin = 2;
int val;


void setup()
{
pinMode(ledpin,OUTPUT);
pinMode(buttonPin,INPUT);
}


void loop()
{
  val=digitalRead(buttonPin);
  if(val==HIGH)
{
  digitalWrite(ledpin,HIGH);
  delay(1000);
  digitalWrite(ledpin,LOW);
  delay(1000);
}
```

```
else{digitalWrite(ledpin,LOW);}
}
```

كيف سيعمل الكود ؟

1 – بدئنا في تعريف المتغيرات

```
int ledpin = 13;
int buttonPin = 2;
int val;
```

2 – جعلنا الاردوينو يفهم ان المخرج 2 هو الذي يعطينا اشارات الدخل (HIGH,LOW) لادخالهم للتحكم في مخرج 13

```
void setup()
{
pinMode(ledpin,OUTPUT);
pinMode(buttonPin,INPUT);
}
```

3 – قمنا بتعريف المتغير val على اساس انه حاضنة للجهد الذي يخرج من 2 فهو يقوم بتخزين القيمة الجديدة , اذا كانت HIGH=1 و LOW=0

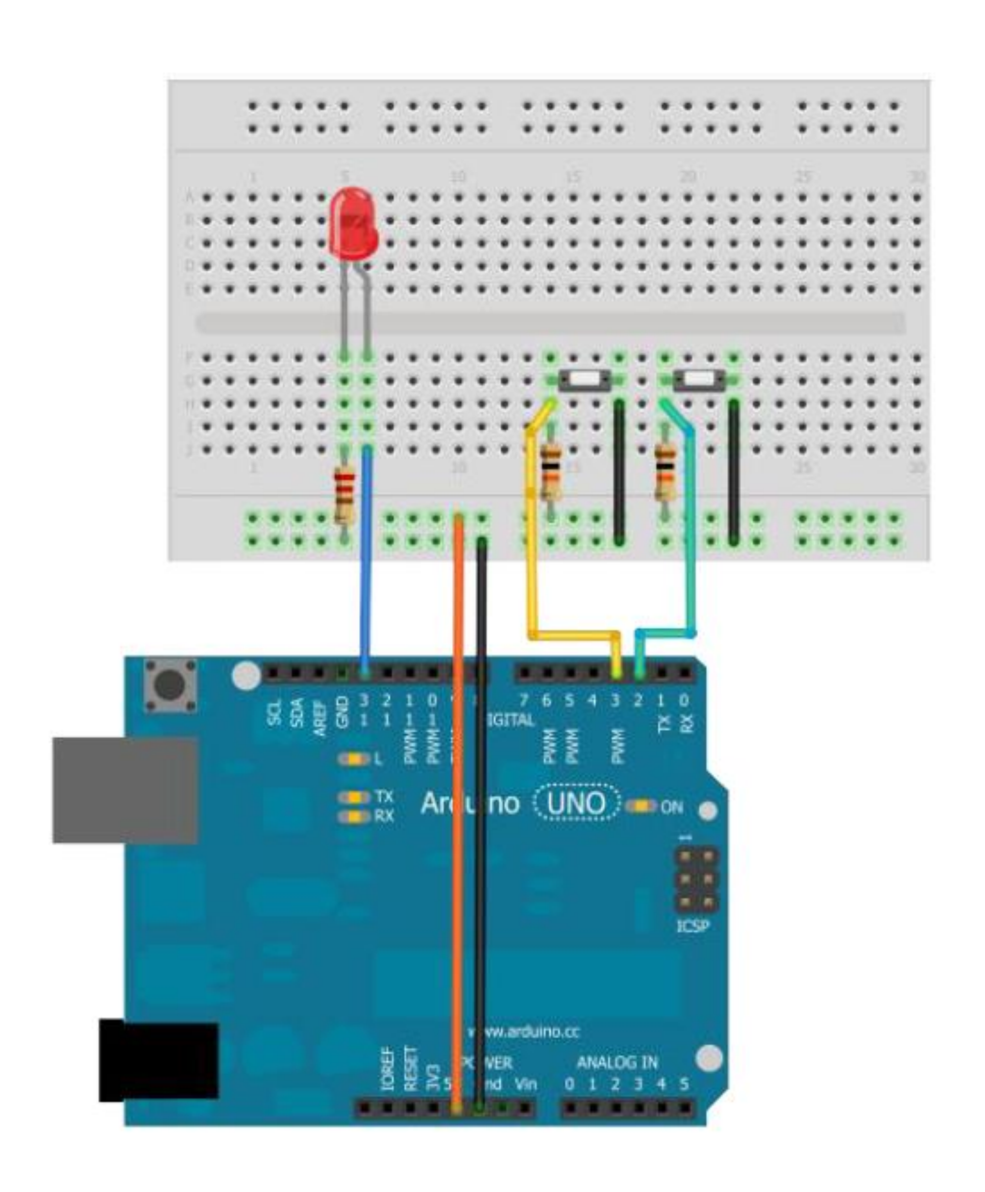
```
void loop()
{
  val=digitalRead(buttonPin);
  if(val==HIGH)
{
  digitalWrite(ledpin,HIGH);
  delay(1000);
```

```
  digitalWrite(ledpin,LOW);
  delay(1000);
}
else{digitalWrite(ledpin,LOW);}
}
```

وبعدها قمنا بادخال الشرط (if - else) لمقارنة قيمة مخرج 2 اذا ضغط المستخدم المفتاح فسيضيئ المصباح , عدا عن ذلك سيبقى منطفئ.

- المثال الثالث : تشغيل الدايود الضوئي LED باستخدام مفتاحين واحد من اجل التشغيل والثاني من اجل الاغلاق

| القطع المطلوبة | |
|---|---|
| LED | 560 ohm resistor |
| Push button | 10 kohm resistor |
| Arduino uno | Bread board |
| Power tool (اختياري) | Wires |



## الكود

```
int ledpin = 13;
int inputpin1 = 2;
int inputpin2 = 3;

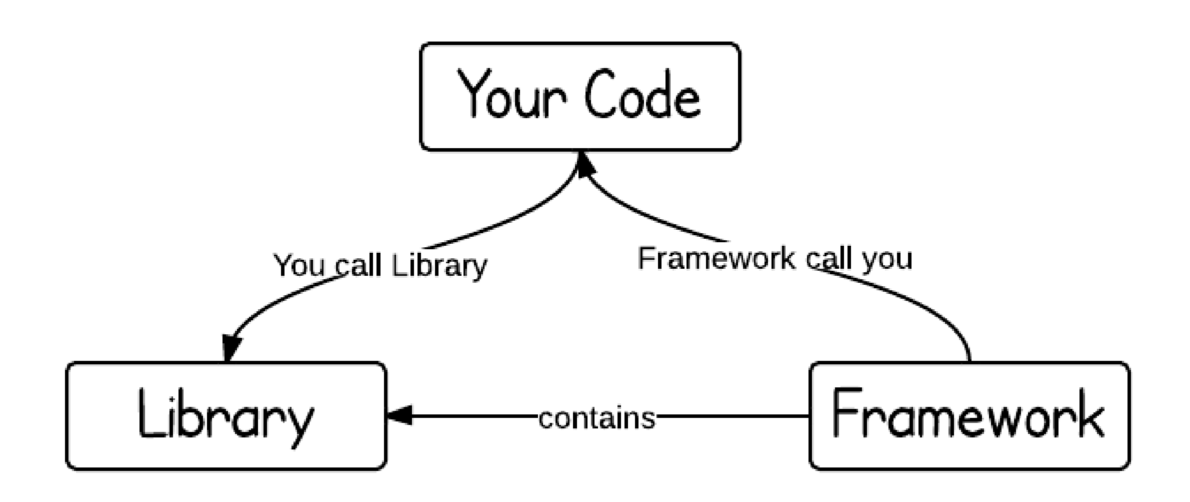
void setup() {

  pinMode(ledpin,OUTPUT);
  pinMode(inputpin1,INPUT);
  pinMode(inputpin2,INPUT);

}

void loop() {
  if(digitalRead(inputpin1)==HIGH)
  {  digitalWrite(ledpin,LOW);  }
else if (digitalRead(inputpin2)==HIGH)
{   digitalWrite(ledpin,HIGH);   }
}
```

## المكتبات Libraries

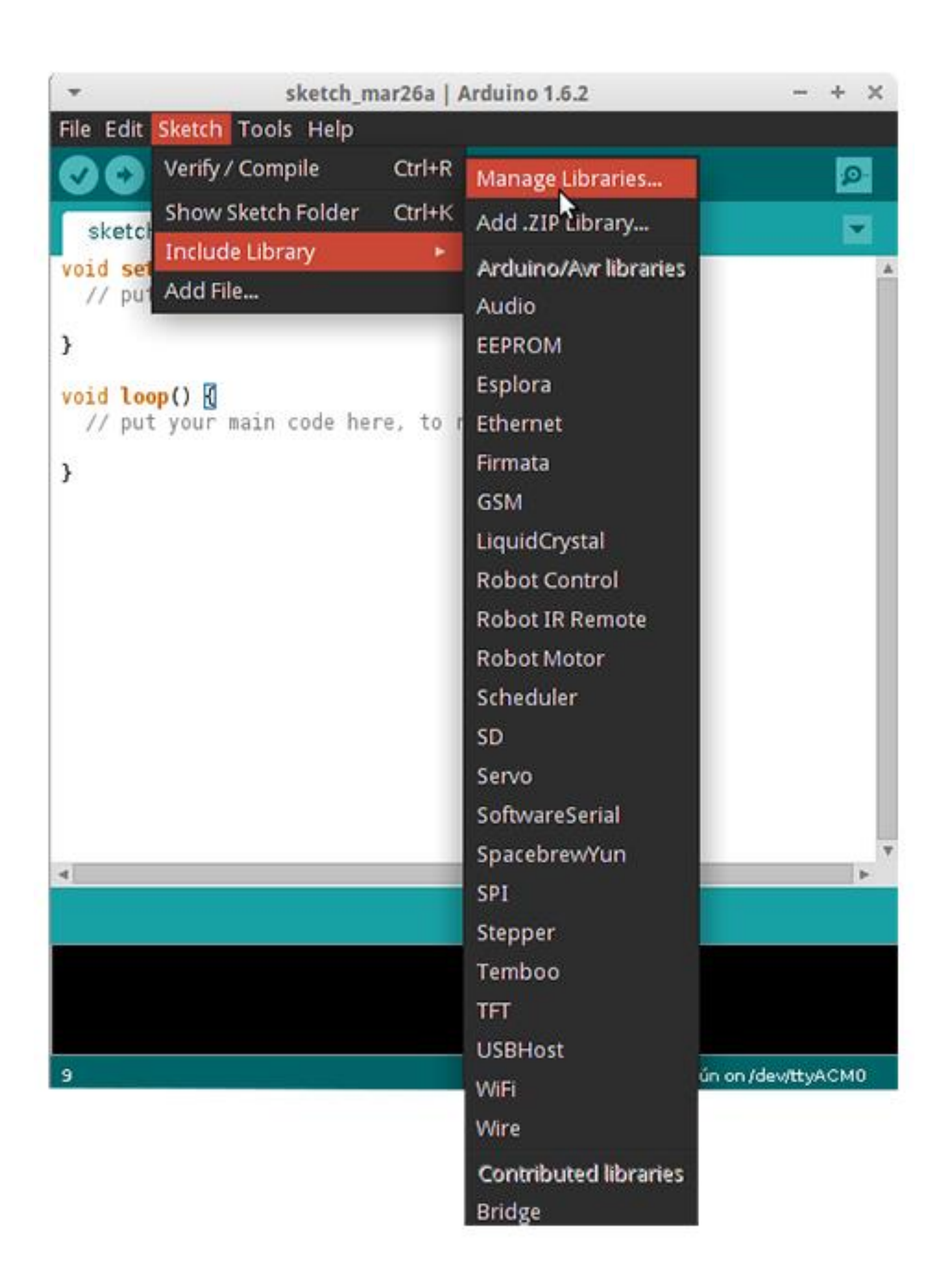



ان برنامج الاردوينو Arduino IDE عبارة عن مكتبات في كل مكتبة فيها مجموعة اوامر Functions مثل العديد من لغات البرمجة , توفر المكتبات وظائف إضافية لاستخدامها في المشاريع مثل الحساسات والمحركات والشاشات وغيرها , مثلا عندما نسخدم محركات السيرفو يجب تنصيب مكتبة محركات السيرفو داخل البرنامج اذا لم ننصب المكتبة فسيطعي البرنامج Error لان اوامر الكود غير معرفة في البرنامج , فهذه من المشاكل التي يغفل عنها المبتدئين في الاردوينو , ونقوم بتعريف المكتبة من خلال : Sketch > Include Library

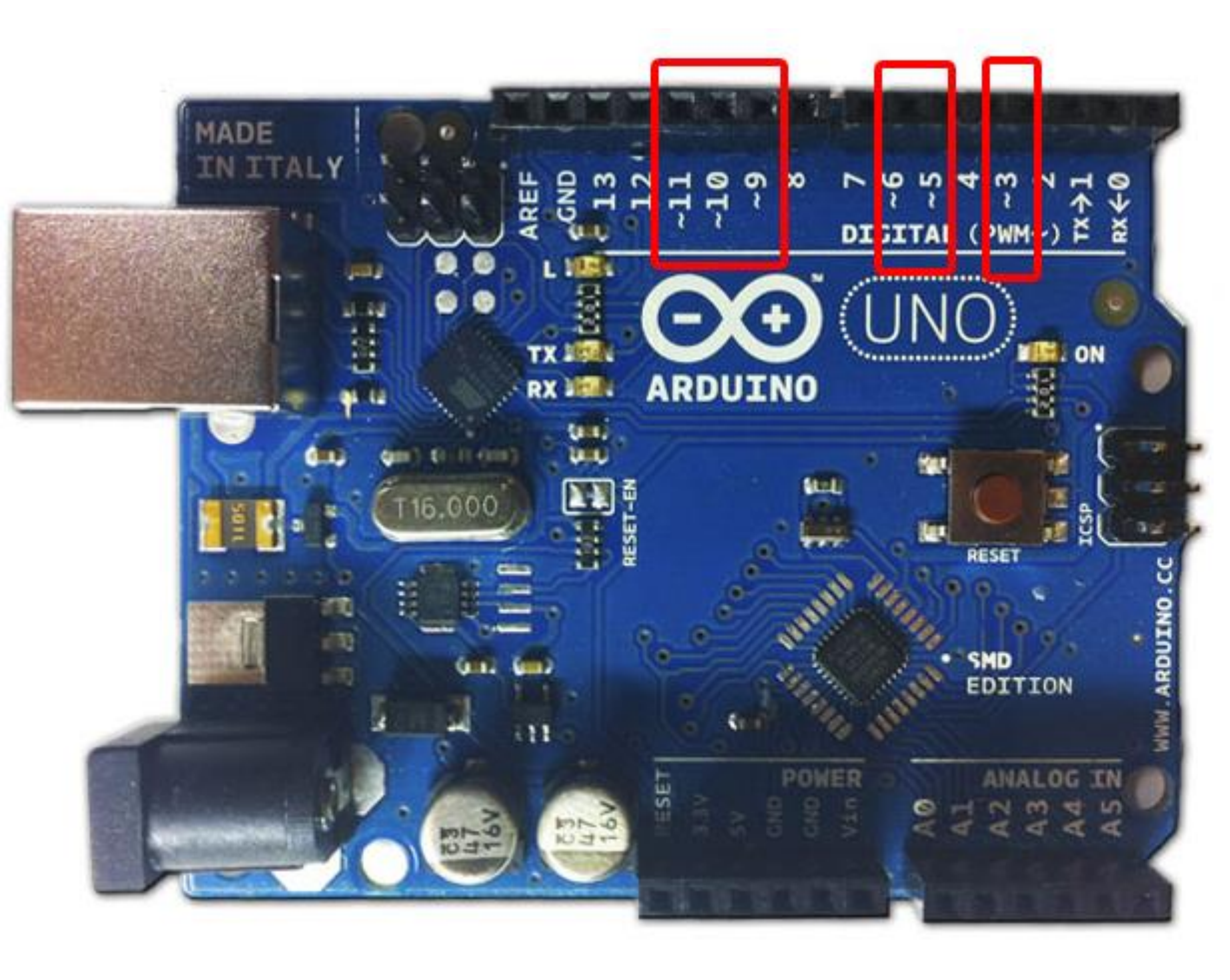
يتم تثبيت عدد من المكتبات باستخدام IDE ولكن يمكنك أيضًا تنزيل أو إنشاء مكتبة خاصة بك , بعض المكتبات الشائعة التوفر على الموقع الرسمي للاردوينو :

| | | |
|---|---|---|
| Servo | LiquidCrystal | GSM |
| WiFi | Audio | Robot |
| Messenger | XBee | SerialControl |

طريقة تنصيب المكتبة داخل البرنامج :



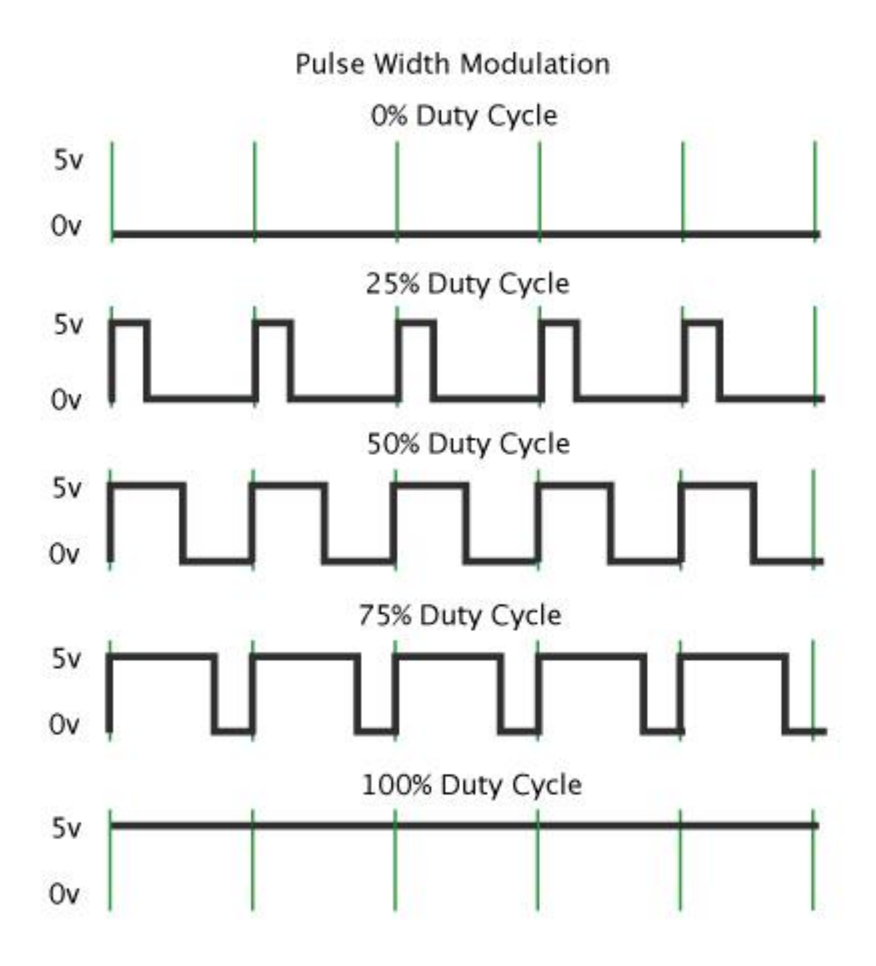
## استخدام PWM




قمنا في دراسة تعديل عرض النبضة والتعرف على الخصائص التي يوفرها , في هذا الدرس سوف نقوم في تطبيق ما تعلمناه من دروس نظرية الى درس تطبيقي في استخدام هذه الخاصية .

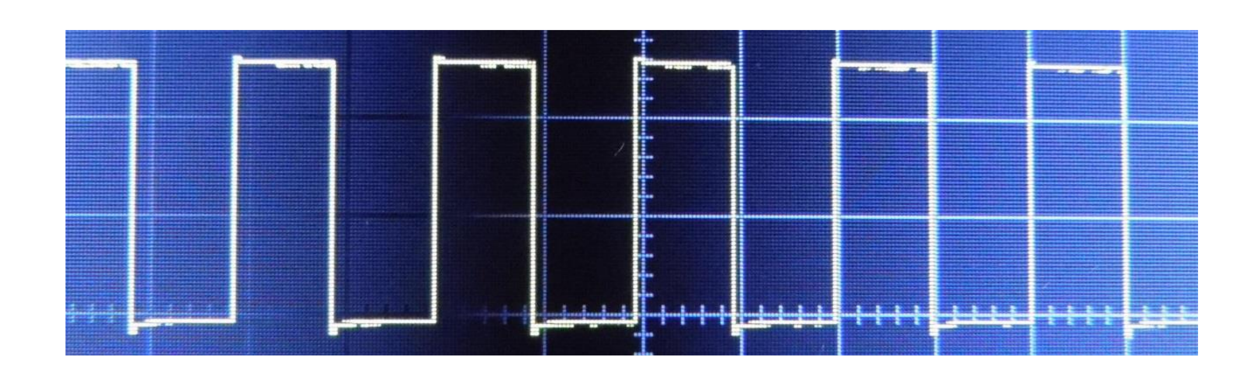
PWM هو مختصر لـ Pulse Width Modulation اي تحميل عرض النبضة.
فكرة ال PWM هي إعطاء نبضة تحتوي على جزء مساوي لقيمة الفولتية للآردوينو و جزء آخر مساوي للصفر فولت . بتغيير نسبة عرض الجزء الموجب إلى عرض

النبضة (Duty Cycle) سيتغير معدل الفولتية لتلك النبضة. أي إن نسبة الجزء الموجب إذا كانت 0% نسبة إلى عرض النبضة فأن قيمة معدل الفولتية ستكون صفر فولت في حين لو كانت هذه النسبة 100% فسيكون معدل الفولتية مساوي للفولتية العليا لمخرج الآردوينو.

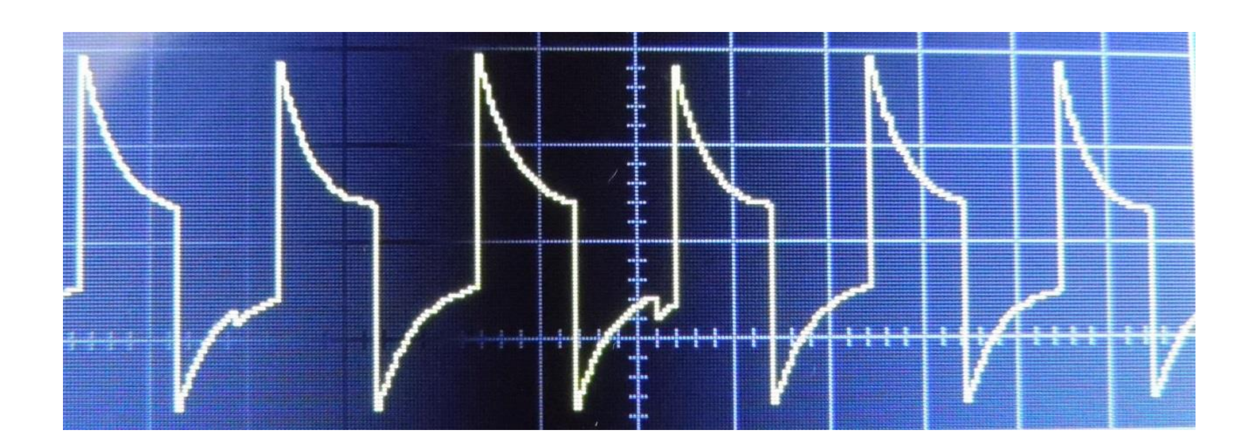



كما تلاحظ من الرسم أعلاه, يمكنك الآن الحصول على معدل فولتية يتراوح بين صفر فولت و الفولتية العليا لمخرج الاردوينو. لكن هذه الفولتية لن تكون بشكل DC لذلك يجب الانتباه إلى تأثيراتها على الأحمال وخصوصا الأحمال التي تحتوي على ملفات مثل المحركات . سنقوم الآن بتسليط مختلف أنواع الأحمال على موجة PWM بنسبة 50% Duty Cycle لمعرفة تأثيرها على الموجة.

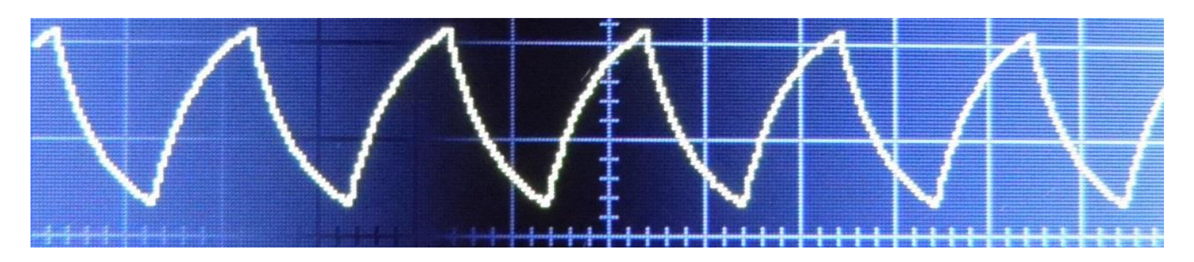
1 - مقاومة : لا تؤثر المقاومة على شكل الموجة الخارجة.



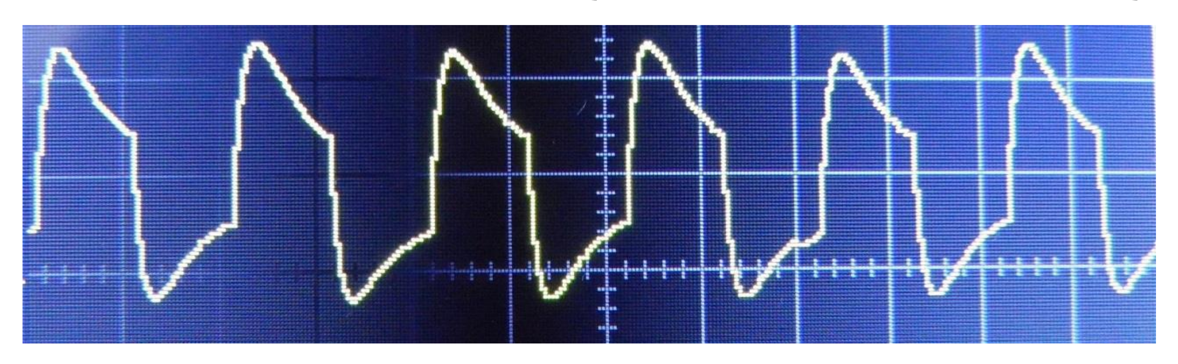
2 - محرك DC : ان الأحمال الحثية هي الأكثر تأثراً و قت تسبب ضرر أيضاً لدائرة الـPWM بسبب الفولتية المحتثة المعاكسة المتولدة في المحرك و التي قد تكون كبيرة بما فيه الكفاية للتسبب بأضرار لدائرة PWM. يمكن التخلص من هذه الفولتية السالبة بإضافة ثنائي يكون الأنود فيه مربوط للقطب السالب للمصدر و الكاثود مربوط إلى مخرج PWM بحيث إن التيار المحتث المعاكس سيمر عن طريق الدايود و لا يسبب ضرر في دائرة PWM .



3 – المكثف : يحاول المكثف دائما الحفاظ على مستوى الفولتية بدون تغيير, لذلك فهو يقلل من التغيير المفاجيء للفولتية الذي يحدث أثناء توليد موجة PWM.



4 - محرك مع مكثف على التوازي. نلاحظ من الشكل التالي إن ربط مكثف بالتوازي مع المحرك يساعد على تقليل التغير السريع للفولتية.



اما كيفية السيطرة على PWM في الآردوينو , يرمز ل PWM على لوحة الاردوينو بالرمز ( ~ ) يحتوي الآردوينو على اكثر من مخرج من الممكن استخدامها للحصول على موجات PWM. يستخدم الأمر analogWrite للحصول على هذه الموجات على المخرج المحدد بالأمر وبـ Duty Cycle المحددة في الأمر أيضاً.

يمكن تحديد Duty Cycle تتراوح بين 0 و 255, حيث تمثل القيمة 255 النسبة 100% أي إن فولتية المخرج المحدد ستكون مساوية لفولتية الاردوينو في حين تثمل القيمة 0 النسبة 0% أي إن فولتية المخرج ستكون صفر فولت. للحصول على فولتية تساوي نصف فولتية المصدر على المخرج رقم 3 نستخدم الأمر analogWrite.

السيطرة على LED , قد يتوهم البعض إن استخدام PWM في السيطرة على إضاءة LED هي ناتجة عن تغير معدل الفولتية الخارجة من الاردوينو و هذا المفهوم خاطئ لكون استجابة الLED لتغير الفولتية قليل جدا و يكون في مجال فولتية ضيق. الا ان

اطفاء و تشغيل الـLED بسرعة عالية يوهم العين البشرية بأن اضاءة الـLED تخفت أو تزداد حسب الـDuty Cycle. أي إن زيادة فترة تشغيل الـLED نسبة إلى عرض النبضة ستوهم العين البشرية بزيادة توهج الـLED. اذا فالـLED يكون في حالة انطفاء و اشتغال سريعة جدا ولا يكون في حالة تشغيل خافت.

لنقم الآن بكتابة برنامج بسيط يقوم بقياس الفولتية على المدخل رقم A0 و على أساس قيمتها يقوم بالسيطرة على LED مربوط إلى المخرج رقم 3.

```
void setup() {

}

void loop() {
  int VI;
  int VO;
  VI = analogRead(0);
  VO = VI/4;
  analogWrite(3,VO);

}
```

يقوم البرنامج في البداية بقياس الفولتية على المدخل A0, ثم يقوم بتقسيم الفولتية على 4 لكون ناتج قياس الفولتية يكون من 0 الى 1023 في حين ان تحديد Duty Cycle يكون من 0 إلى 255. يقوم البرنامج بعد ذلك بتوليد إشارة PWM على المخرج رقم 3.

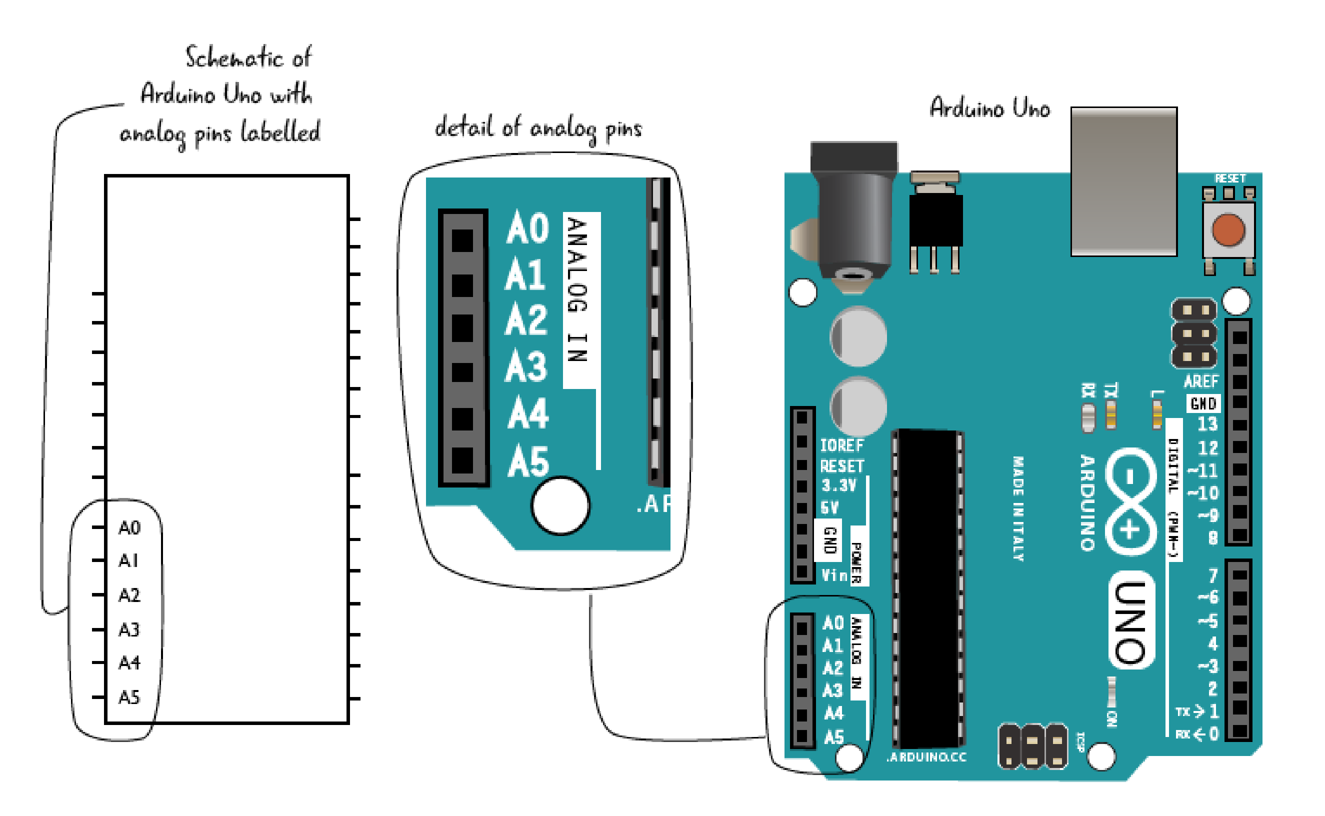
مثال آخر للتحكم في اضاءة LED باستخدام مقاومة متغيرة :

```
int ledPin = 9; // LED connected to digital pin 9
int analogPin = 3; // potentiometer connected to analog pin 3
int val = 0; // variable to store the read value

void setup() {
```

```
  pinMode(ledPin, OUTPUT); // sets the pin as output
}

void loop() {
  val = analogRead(analogPin); // read the input pin
  analogWrite(ledPin, (val / 4)); // analogRead values go from 0
to 1023,
    // analogWrite values from 0 to 255
}
```




يحتوي الآردوينو على نقاط ارتباط من الممكن ربطها إلى دائرة ( Analog To Digital Converter) داخل المعالج الدقيق تقوم بقياس قيمة الفولتية المسلطة على هذا المدخل. تقوم دوائر (ADC) بقياس قيمة الفولتية المطلوبة نسبة إلى نقطتين مرجعيتين. أي أن هذه الدائرة لا تستطيع قياس الفولتية بشكل مطلق. تقاس دقة دائرة التحويل بعدد البتات (bit) الناتجة من عملية التحويل حيث إن زيادة عدد البتات الناتجة من العملية تعطينا دقة اكثر , لمعرفة السبب وراء ذلك علينا فهم الطريقة التي تتبعها هذه الدائرة للعمل.

كيف تعمل دوائر التحويل من خطي الى ثنائي ؟

لفهم كيفية عمل دوائر (ADC) سنقوم بأخذ معطيات دوائر (ADC) الموجودة على الآردوينو كمثال. يتكون ناتج دوائر ADC في الآردوينو من 10) bit توجد موديلات

معينة من الآردوينو تحتوي على ADC بدقة 12 بت). أي إن ناتج عملية التحويل ممكن أن يتراوح بين 0 و 1023 (0000000000 , 1111111111 بالنظام الثنائي). تقوم دائرة التحويل بتقسيم الفولتية المرجعية إلى 1023 جزء. تكون فولتية المرجعية افتراضياً هي فولتية الدخول أي إن فولتية المرجع الأدنى هي 0 فولت و المرجع الأعلى 5 فولت. بتقسيمها إلى 1023 جزء, ستكون فولتية الجزء الواحد (الخطوة الواحدة) 0.00488 فولت. لقياس الفولتية الداخلة, تقوم الدائرة بتشغيل عداد يحسب عدد الخطوات التي تحتاجها الدائرة للوصول إلى فولتية تكون مساوية أو اكبر من الفولتية المطلوب قياسها, عند الوصول إلى هذه الفولتية, تعطي الدائرة قيمة العداد كمقياس لنسبة الفولتية الداخلة إلى فولتيات المرجعية.

لتوضيح فكرة دقة الدائرة (Resolution), لنأخذ دائرتين بدقتين مختلفة و نقارن بينهما. الدائرة الأولى يتكون ناتجها من أربعة بتات (bit resolution4) و الثانية من عشر بتات (bit resolution10). فولتية المرجعية لكلتا الدائرتين متطابقة (0-5) فولت و الفولتية الداخلة المطلوب قياسها هي 2.75 فولت.

ستقوم الدائرة الأولى بتقسيم فولتية المرجعية إلى 32 جزء. أي إن قيمة الخطوة الواحدة ستكون 0.15625 فولت. سيبدأ العداد بالتزايد ويصل حتى الخطوة رقم 18 حيث ستصبح الفولتية المقارنة مع الفولتية الداخلة اكبر فيتوقف العداد وتكون النتيجة 18. بمعرفة الفولتية المرجعية, نستطيع الآن أن نحسب قيمة الفلوتية التي تم قياسها و هي ((32/18)*(5-0)) حيث ستكون النتيجة 2.8125 فولت.

لنقم الآن بإعادة الحسابات للدائرة ذات الدقة 10 بت. تكون فولتية الخطوة الواحدة 0.00488. سيبدأ العداد بالزيادة حتى يصل الخطوة رقم 563, حيث ستصبح الفولتية المقارنة مع الفولتية الداخلة اكبر و سيكون ناتج التحويل هو 563. بإعادة حساب الفولتية ((1023/563)*(0-5)) تكون الفولتية المحسوبة 2.75171 فولت.

لاحظ الآن الفرق بين النتيجتين عن الفولتية الداخلة. تحيد قراءة الدائرة الأولي عن القيمة الحقيقية بـ (2.8125 - 2.75) أي 0.0625 فولت (62.5 ملي فولت) في حين تحيد الفولتية ألمقاسه عبر الدائرة الثانية (2.75171 - 2.75) أي 0.00171 فولت (1.71 ملي فولت) .

تحتاج دوائر ADC إلى معالجات سريعة جدا لزيادة دقة القياس وإلا فإن الزمن الذي ستستغرقه عملية القياس سيكون طويلا. تزداد أهمية استخدام دوائر بدقة أعلى مع زيادة الفرق بين فولتيتي المرجعية.

- إعدادات ADC الموجود في الآردوينو.

اصبح من الواضح الآن إن لعمل أي ADC فهو بحاجة إلى معرفة فولتيات المرجعية ليقارن الفولتية الداخلة معها, و إن ناتج العملية يكون نسبة الى فولتية المرجعية. إضافة إلى ذلك يمكنك تحديد دقة ADC الموجود في الآردوينو في حال كنت تحتاج إلى دقة اقل و وقت قياس اسرع. لتحديد فولتية المرجعية الأعلى للآردوينو نستخدم الأمر analogReference(), يمكنك الاختيار من واحد من المرجعيات التالية :

- DEFAULT , تكون فولتية المرجعية الأعلى هي فولتية التغذية للوح التطوير (5 فولت او 3.3 فولت حسب الموديل )
- INTERNAL , وهي فولتية مرجعية داخلية موجودة في الآردوينو (1.1 فولت للآردوينو موديل ATmega168 او ATmega328 و 2.56 فولت للموديل ATmega8 ) لا يمكن استخدام هذا الاختيار مع الآردوينوMega
- INTERNAL1V1 , فولتية مرجعية داخلية في الآردوينو Mega فقط بحيث تكون فولتية المرجعية العليا 1.1 فولت.
- INTERNAL2V56 , فولتية مرجعية داخلية في الآردوينو Mega فقط , تكون فولتية المرجعية العليا 2.56 فولت.
- EXTERNAL , فولتية مرجعية عليا خارجية تربط إلى نقطة الارتباط AREF وانتبه إلى إن هذه الفولتية يجب أن تكون بين 0 و 5 فولت. و يجب أن تقوم بوضع هذا الأمر قبل أي أمر لقراءة الفولتية لأن ذلك قد يسبب التلف للآردوينو.

أما لتحديد دقة القياس فيمكن تحديد عدد بتات الناتج عن طريق الأمر analogReadResolution(), مع تحديد عدد البتات داخل الأقواس بين 0 و 16.

لاحظ انك تستطيع تقليل دقة القياس لكن لا يمكنك زيادتها عن القدرة ألتصميميه للآردوينو أما في حال قمت باختيار رقم أعلى من القدرة التصميمية فأن الناتج لن يزيد عن القيمة العليا المصمم لها لكنه سيقوم بإضافة أصفار للوصول إلي العدد المطلوب من البتات.

- كيفية قياس الفولتية الداخلة

لقياس الفولتية الداخلة على احد أطراف الارتباط, يمكنك استخدام الأمر .analogRead()يجب تحديد الطرف المطلوب القياس منه, لقياس الفولتية على الطرف A0 نكتب الأمر.analogRead(A0)

كما ذكرنا سابقاً فإن ناتج عملية القياس هو نسبة الفولتية المقاسه إلى فولتية المرجعية. لمعرفة قيمة الفولتية الداخلة بالفولت نقوم بقسمة الناتج على عدد خطوات ADC ثم نقوم بضربها في قيمة فولتية المرجعية الأعلى. مثال, لنفرض انك حصلت على ناتج 500 وكانت دقة المقياس 10 بت و فولتية المرجعية الأعلى هي 5 فولت فإن الفولتية الداخلة ستكون ((1023/500)*5) سيكون الناتج 2.44 فولت.

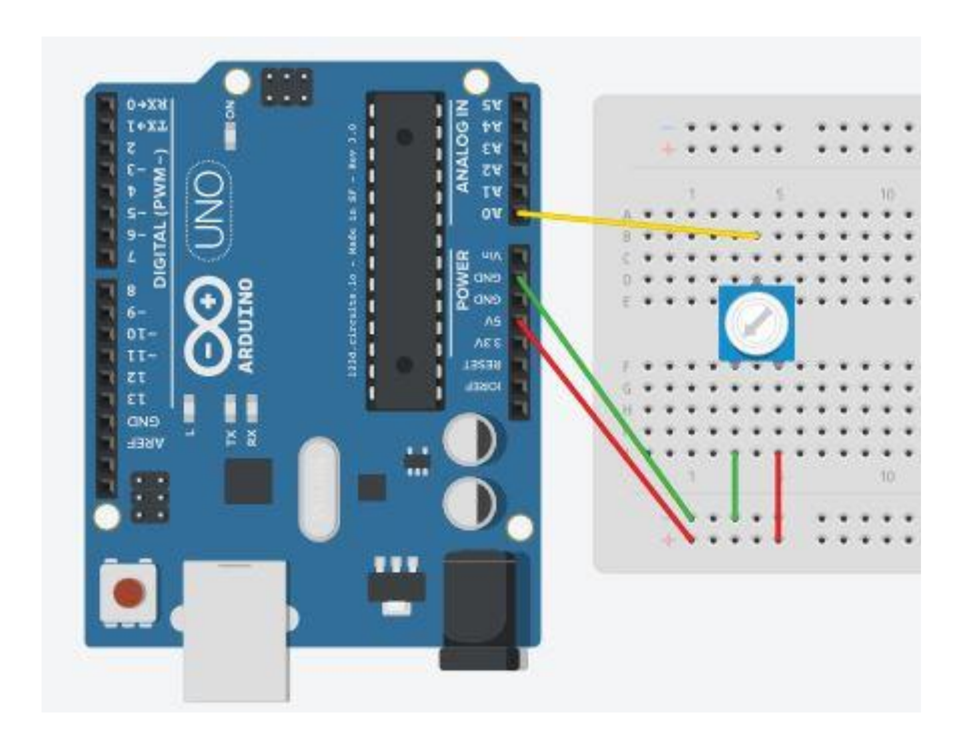
لنبدأ الآن بعمل تجربة عملية لتوضيح الفكرة اكثر وهي تشغيل الآردوينو كمقياس فولتية (فولتميتر) .

سنقوم الآن بكتابة برنامج يقوم بقياس الفولتية الداخلة إلى نقطة الارتباط A0 و عرض الفولتية الناتجة على شاشة الحاسوب. نربط طرفي مقاومة متغيرة إلى V5 و GND في الآردوينو و نربط الطرف الأوسط للمقاومة إلى الطرف A0 في الآردوينو. ستتغير الفولتية على طرف المقاومة الأوسط بتحريكها بقيمة تتراوح من 0 إلى 5 فولت. لنبدأ الآن بكتابة برنامج الآردوينو.

يقوم البرنامج ببساطة بقراءة ناتج analogRead و استخراج قيمة الفولتية الداخلة بالفولت و عرضها على الحاسوب عن طريق الأمر Serial.println.

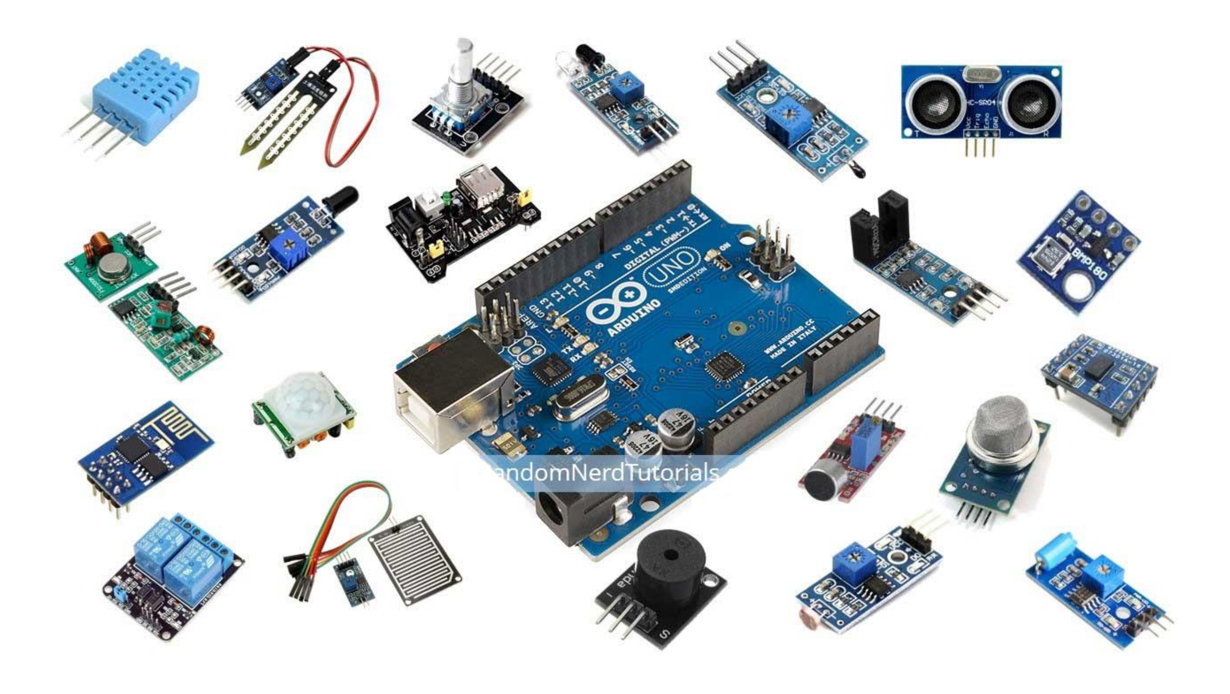
```
void setup() {
Serial.begin(9600);
}
void loop() {
int m;
float r;
```

```
m = analogRead(A0);
r = m*5.0/1023.0;
Serial.println(r);
delay(500);
}
```




تلاحظ هنا وجود نوعين من المتغيرات في البرنامج, الأول من نوع int أي رقم صحيح ونضع فيه القيمة الناتجة من عملية القياس و الثاني من نوع float وهو النوع الذي يحتوي على أعشار. لاحظ أيضاً أهمية كتابة الأرقام 5.0 و 1023.0 ليتم التعامل معها على إنها أرقام تحتوي على أعشار وإلا ستكون النتيجة عدد صحيح . أما delay فهو يستخدم لتأخير الوقت بين قياس و آخر و إلا ستكون القراءات على الحاسوب اسرع مما تستطيع قراءته. قد تلاحظ وجود فرق بسيطة بين قراءات الآردوينو و الفولتميتر يعود إلى وجود نسبة خطأ في كلا الجهازين وهو أمر طبيعي.

# استخدام الحساسات




الحساس أو المجس أو المستشعر هو أداة استشعار، يعمل على كشف الحالة المحيطية الفيزيائية، فمنه ما يقيس درجة الحرارة، ومنه ما يقيس الضغط ومنه ما يقيس الإشعاع ومنه ما يقيس الإلكترونات أو البروتونات. حيث يقوم بتحويل الإشارات الساقطة عليه إلى نبضات كهربائية يمكن قياسها أو عدَّها بواسطة جهاز. بهذا يمكن لنا معرفة شدَّة المؤثر. كما توجد أنواع منه يمكن ربطها بأجهزة حاسوب وعن طريق البرمجة يمكن تكوين صورة عن توزُّع القياسات، كما هو الحال في التصوير بالرنين المغناطيسي الذي يكشف في الإنسان عن أورام. من المجسات أنواع كثيرة بحسب الاستخدام منها في هذا الدرس سوف نبدأ في التعامل مع الحساسات المختلفة التي يوفرها الاردوينو ويستطيع التعامل معها لبناء مشاريع تفاعلية . ( من الضروري مراجعة الكتاب الثالث من السلسلة لفهم آلية تطبيق هذا الدرس )

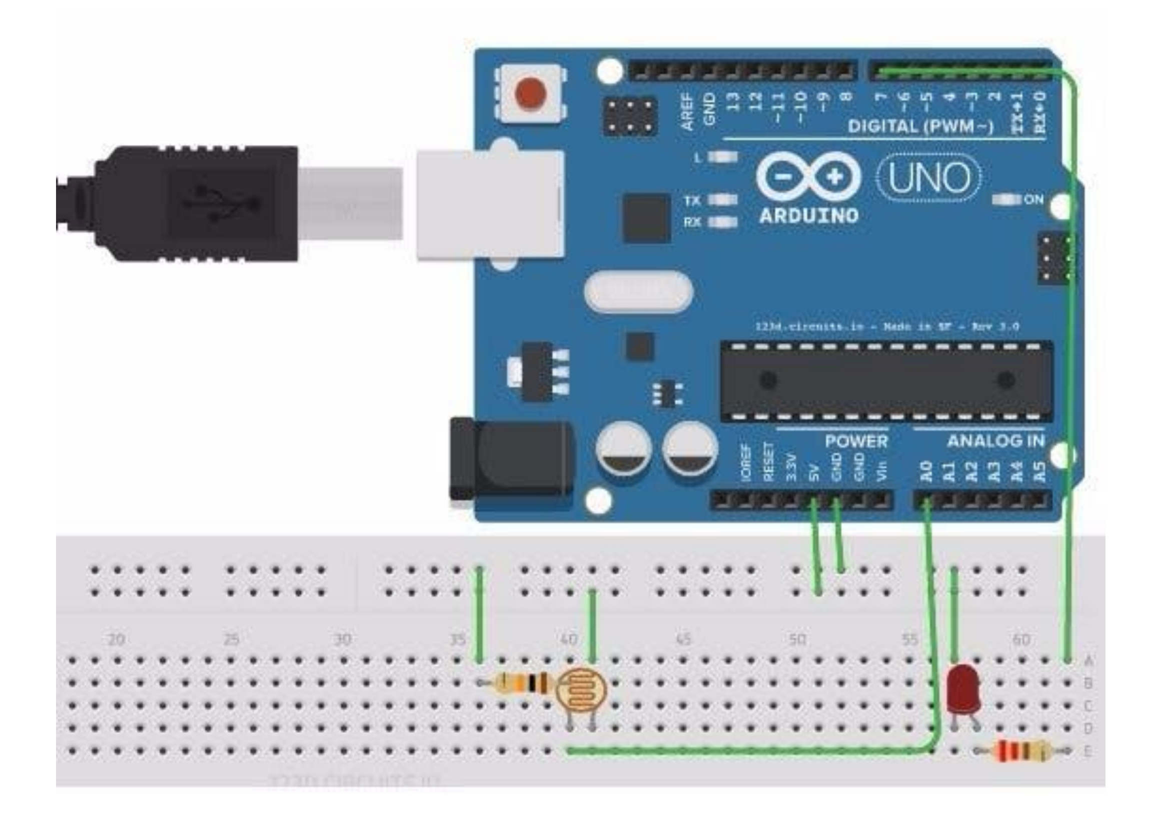
- المثال الرابع : استخدام الحساس الضوئي LDR كمفتاح

في كل مرة تقوم فيها بتغطية السنسور، يتم تشغيل LED أو أيًا كان في حالة إيقاف تشغيله أو إيقاف تشغيله إذا كان مشغلاً.

القطع المطلوبة :

| | |
|---|---|
| LDR | |
| Resistor 10 k ohm | |
| Resistor 221 ohm | |

المخطط :



الكود :

```
const int lamp = 7;
boolean x = true;

void setup() {
  Serial.begin(9600);
  pinMode(lamp , OUTPUT);

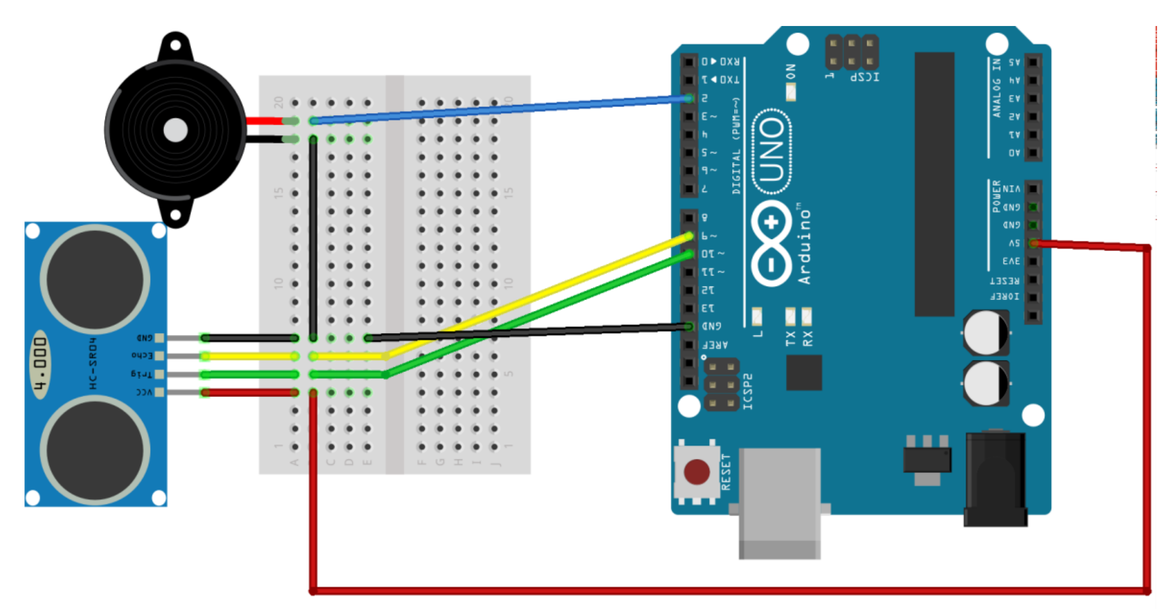
}

void loop() {
  int c = analogRead(A0);
  delay(500);
if ( c<300 && x == true){
  digitalWrite(7,HIGH);
   x = false;
   delay(1000);

}
else if ( c <300 && x == false){
x = true;
digitalWrite(7,LOW);
delay(1000);
}
}
```

▪ المثال الخامس : الكشف عن العقبات و التحذير باستخدام حساس الموجات فوق الصوتية

المخطط :



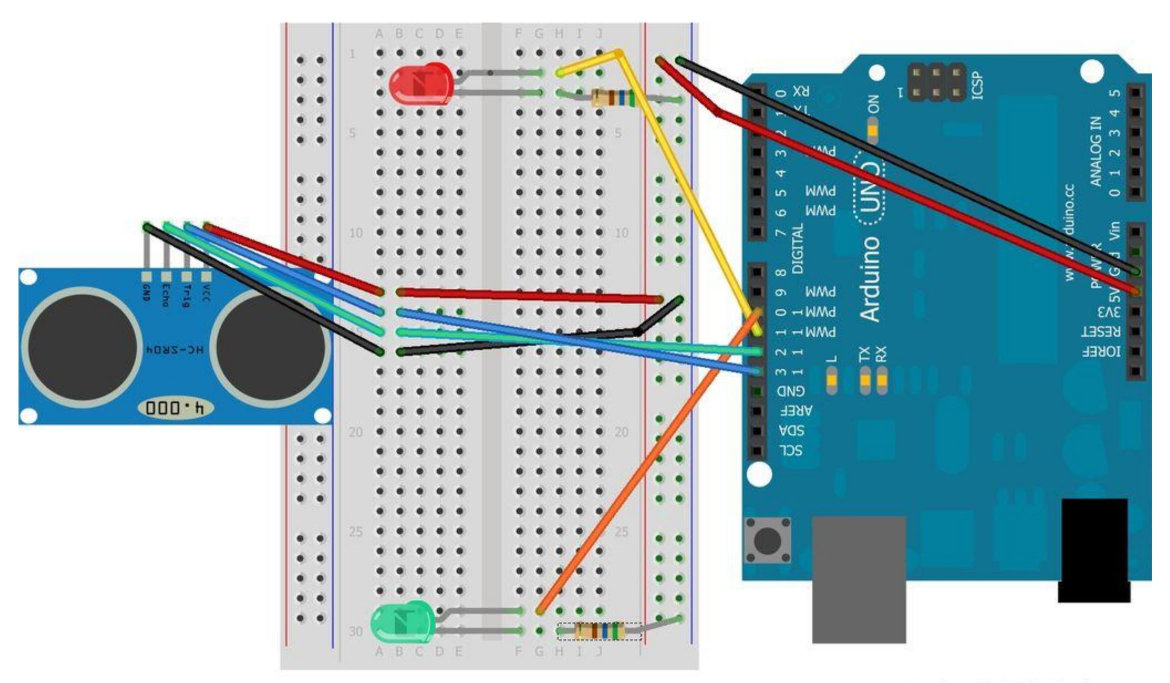

الكود :

```
// Define pins for ultrasonic and buzzer
int const trigPin = 10;
int const echoPin = 9;
int const buzzPin = 2;
void setup()
{
pinMode(trigPin, OUTPUT); // trig pin will have pulses output
pinMode(echoPin, INPUT); // echo pin should be input to get pulse
width
pinMode(buzzPin, OUTPUT); // buzz pin is output to control
buzzering
}
void loop()
{
// Duration will be the input pulse width and distance will be the
distance to the obstacle in centimeters
```

```
int duration, distance;
// Output pulse with 1ms width on trigPin
digitalWrite(trigPin, HIGH);
delay(1);
digitalWrite(trigPin, LOW);
// Measure the pulse input in echo pin
duration = pulseIn(echoPin, HIGH);
// Distance is half the duration devided by 29.1 (from datasheet)
distance = (duration/2) / 29.1;
// if distance less than 0.5 meter and more than 0 (0 or less means
over range)
if (distance <= 50 && distance >= 0) {
// Buzz
digitalWrite(buzzPin, HIGH);
} else {
// Don't buzz
digitalWrite(buzzPin, LOW);
}
// Waiting 60 ms won't hurt any one
delay(60);
}
```

المثال التالي يوضح  التحكم في وميض مصباحين LED باستخدام حساس الامواج فوق الصوتية

المخطط :



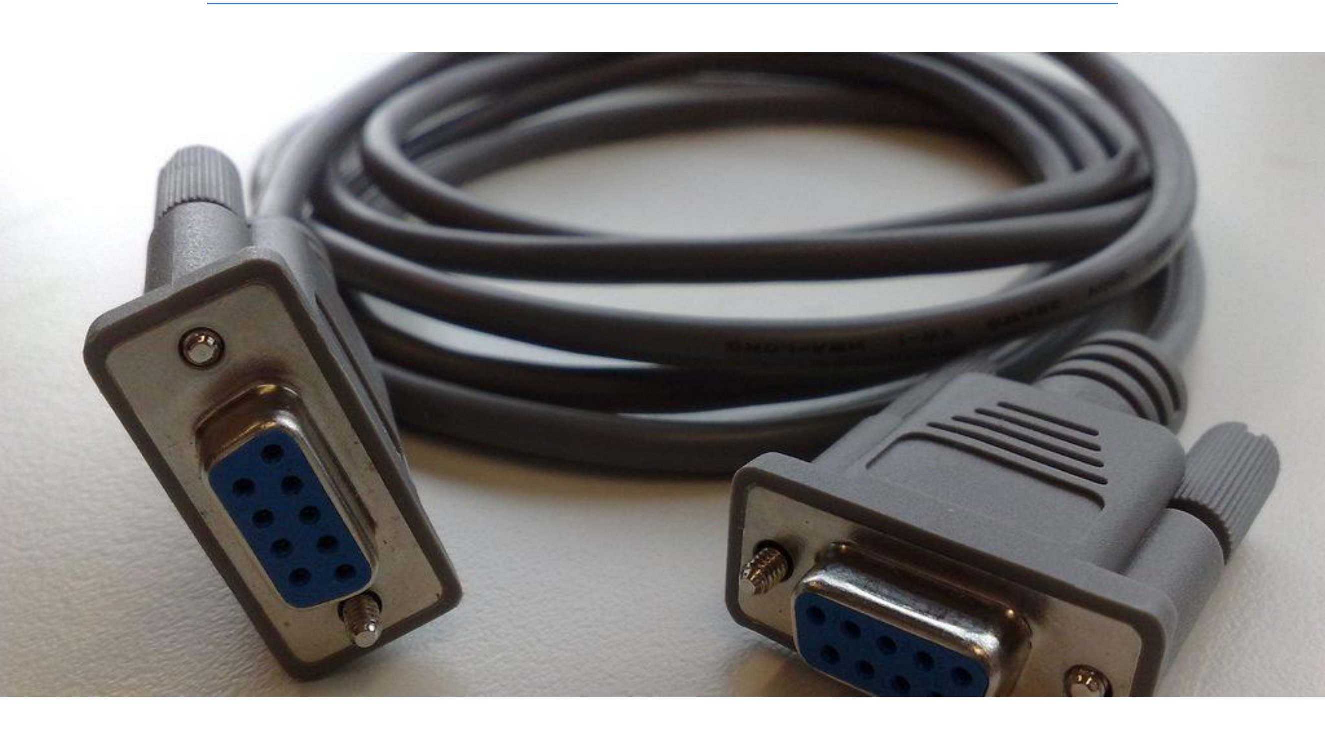
الكود :

```
#define trigPin 13
#define echoPin 12
#define led 11
#define led2 10

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(led, OUTPUT);
  pinMode(led2, OUTPUT);
}
```

```
void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);  // Added this line
  delayMicroseconds(2); // Added this line
  digitalWrite(trigPin, HIGH);
//  delayMicroseconds(1000); - Removed this line
  delayMicroseconds(10); // Added this line
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  if (distance < 4) {  // This is where the LED On/Off happens
    digitalWrite(led,HIGH); // When the Red condition is met, the
Green LED should turn off
  digitalWrite(led2,LOW);
}
  else {
    digitalWrite(led,LOW);
    digitalWrite(led2,HIGH);
  }
  if (distance >= 200 || distance <= 0){
    Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
  }
  delay(500);
}
```

## المنفذ التسلسلي Serial Communication

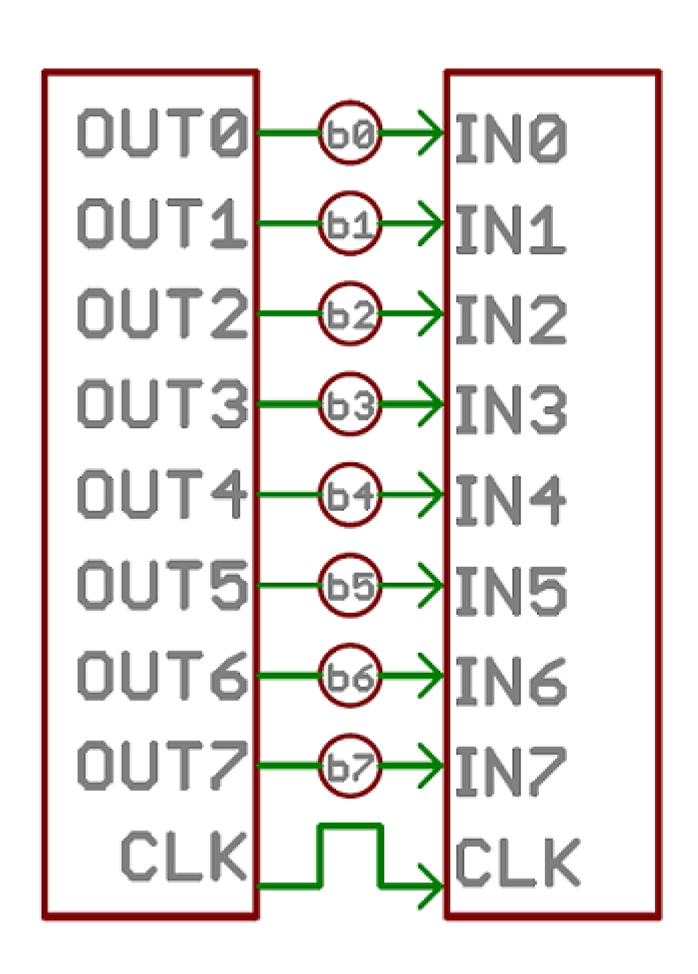


الإلكترونيات المدمجة (Embedded electronics) هي عبارة عن دوائر إلكترونية مترابطة معاً (مُعالجات (processors) أو دوائر متكاملة (integrated circuits)أخرى) لإنشاء نظام تكافلي .(symbiotic system) ولجعل تلك الدوائر تتبادل المعلومات مع بعضها البعض لا بد أن تتشارك في بروتوكول اتصال (communication protocol)مشترك. هناك المئات من بروتوكولات الاتصال تستطيع القيام بمهمة تبادل البيانات تلك، وبوجه عام يُمكن تقسيم تلك البروتوكولات إلى نوعين: تسلسلية (serial) ومتوازية.(parallel).

## الاتصال التسلسلي والاتصال المتوازي

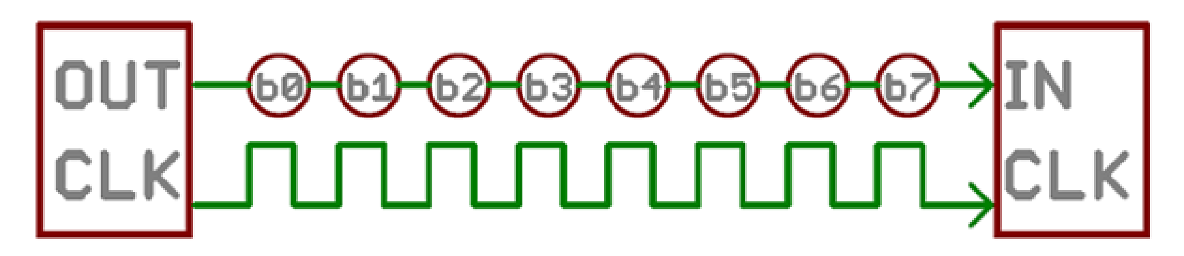
الواجهات المتوازية تقوم بنقل عدة بتات (bits) في نفس الوقت. وهي في العادة تتطلب نواقل (buses) بيانات تستخدم عدد أسلاك يبلغ ثمانية أو ستة عشر أو أكثر، ويتم نقل البيانات في أمواج ضخمة متلاطمة من القيمتين 0 و1

.




ناقل بيانات 8 بت (8-bit data bus) يتم التحكم فيه باستخدام ساعة (clock) بحيث يقوم بنقل بايت (byte) خلال كل نبضة من نبضات الساعة. ويتم استخدام تسعة أسلاك.

أما الواجهات التسلسلية فتقوم بنقل البيانات بت تلو الآخر. هذه الواجهات يُمكن أن تعمل باستخدام عدد قليل من الأسلاك يصل إلى سلك واحد وعادة لا يتجاوز أربعة أسلاك.




مثال على واجهة تسلسلية، تقوم بنقل بت واحد كل نبضة ساعة(clock pulse) ، ولا يتطلب الأمر سوى سلكين فقط.

تخيل أن نوعي الاتصال (التسلسلي والمتوازي) عبارة عن طريق للسيارات: في الاتصال المتوازي يكون الطريق يحتوي على ثمان مسارات(lanes) ، بينما في الاتصال التسلسلي يكون الطريق ريفي ضيق يحتوي على مسارين فقط. على مدى فترة معينة من الزمن يستطيع الطريق الواسع ذو الثمان مسارات إيصال عدد أكبر من السيارات، لكن في الوقت ذاته يؤدي الطريق الريفي ذو المسارين نفس الغرض، ولا يكلف سوى جزء صغير من تكلفة الطريق الواسع.

الاتصال المتوازي له بالتأكيد مميزات، فهو سريع ومستقيم ويسهل تطبيقه. لكنه يتطلب الكثير من خطوط الإدخال (input) والإخراج (I/O) lines)) (output). إذا سبق لك من قبل نقل أحد المشاريع من لوح أردوينو أونو (Arduino Uno) إلى أردوينو ميجا (Mega) فستدرك أن خطوط الإدخال والإخراج الخاصة بالمتحكم الدقيق (microprocessor) قليلة ولها أهمية كبيرة. لذلك نلجأ غالباً لاستخدام الاتصال التسلسلي مضحين ببعض السرعة في سبيل توفير بعض المنافذ.

## الاتصال التسلسلي غير المتزامن(Asynchronous Serial)

على مدى سنوات عديدة تم ابتكار الكثير من بروتوكولات الاتصال التسلسلي لتناسب الاحتياجات الخاصة بالأنظمة المدمجة. الناقل التسلسلي العالمي USB (universal serial bus)، والإيثرنت (Ethernet) هما مثالان على الواجهات التسلسلية الأكثر شهرة في عالم الكمبيوتر. هناك أيضاً واجهات تسلسلية أخرى ذات شهرة كبيرة مثل SPI و$I^2C$ والاتصال التسلسلي القياسي الذي سنتحدث عنه في هذا الدرس. كل من هذه الواجهات التسلسلية يُمكن تقسيمها إلى فئتين: متزامنة (synchronous)، وغير متزامنة (asynchronous).

دائماً ما تحتوي الواجهات التسلسلية المتزامنة على إشارة ساعة بالإضافة إلى خط (خطوط) البيانات، لذلك فإن جميع الأجهزة المتصلة بناقل تسلسلي متزامن تتشارك في ساعة مشتركة. وهذا يساعد في تبسيط وتسريع النقل التسلسلي، ولكنه يتطلب سلك واحد إضافي على الأقل بين الأجهزة المتصلة. تتضمن الأمثلة على الواجهات المتزامنة SPIو$I^2C$ .

أما في الواجهات غير المتزامنة، يتم نقل البيانات بدون الاستعانة بإشارة ساعة خارجية. وهذه الطريقة هي الأمثل لتقليل عدد الأسلاك المطلوبة ومنافذ الدخل والخرج المستخدمة، ولكن هذا يعني الحاجة لبذل المزيد من المجهود لنقل واستقبال البيانات بصورة موثوقة. البروتوكول التسلسلي الذي سنناقشه في هذا الدرس هو الشكل الأكثر شيوعاً للاتصال التسلسلي غير المتزامن. فهو منتشر لدرجة أنه عندما يقول أحد ما "اتصال تسلسلي" فإنه في الغالب يقصد ذلك البروتوكول.

بروتوكول الاتصال التسلسلي الذي لا يتضمن ساعة والذي سنناقشه خلال هذا الدرس يُستخدم على مدى واسع في الإلكترونيات المدمجة. إذا كنت تتطلع لإضافة وحدة GPS أو Bluetooth أو XBee أو شاشات LCD تسلسلية أو العديد من الأجهزة الخارجية إلى مشروعك فستحتاج للتعرف بشكل أكبر على الاتصال التسلسلي.

## قواعد الاتصال التسلسلي

بروتوكول الاتصال التسلسلي الغير متزامن يتضمن عدداً من القواعد والآليات مدمجة به تساعد في ضمان نقل البيانات بشكل سليم وخالي من الاخطاء. هذه الآليات (التي نستعين بها لعدم استخدام إشارة ساعة خارجية) هي:

- بتات البيانات(Data bits)
- بتات المزامنة(Synchronization bits)
- بتات التكافؤ(Parity bits)
- معدل بود(Baud rate)

بسبب تنوع آليات إرسال الإشارات تلك فليست هناك طريقة واحدة معينة لإرسال البيانات تسلسلياً، بل البروتوكول قابل لإعادة التشكيل بشكل كبير. الجزء المهم هنا هو التأكد من أن كلا الجهازين المتصلين بالناقل التسلسلي مُعدان لاستخدام نفس البروتوكولات بالضبط.

### معدل بود (Baud rate)

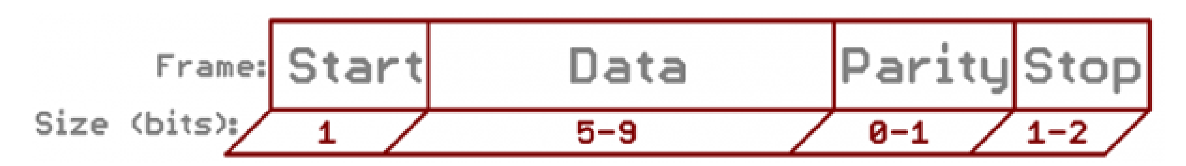
يُحدد معدل بود سرعة إرسال البيانات عبر الخط التسلسلي، وغالباً ما يُعبر عنه بوحدة بت/ثانية .(bps) وإذا قمنا بقلب معدل بود نحصل على المدة المُستغرقة لنقل بت واحد فقط. وهذه القيمة تحدد المدة التي يُبقي فيها المُرسل الخط التسلسلي في حالة مرتفعة (high)/منخفضة.(low)

معدل بود يُمكن أن يكون أي قيمة مُمكنة. والشيء الوحيد المطلوب هو أن يعمل كلا الجهازين بنفس المعدل. من معدلات بود الأكثر انتشاراً –وخاصة في الأجهزة البسيطة التي لا تتطلب سرعة كبيرة- هو 9600 .bpsومن المعدلات القياسية الأخرى 1200 و2400 و4800 و19200 و34800 و57600 و115200.

كلما ارتفعت قيمة مُعدل بود زادت سرعة إرسال/استقبال البيانات، لكن هناك حدود للسرعة التي يُمكن عبرها تقل البيانات. فغالباً لن ترى سرعات تتجاوز 115200. فزيادة السرعة بدرجة كبيرة للغاية يُسبب حدوث أخطاء في الطرف المستقبل لأن السعة ومدد أخذ العيّنات (sampling periods) لا تستطيع التكيف مع ذلك.

## تأطير البيانات(Framing the data)

كل مجموعة (block) عادة ما تكون بايت) من البيانات يتم نقلها في الواقع عبر حزمة (packet) أو إطار (frame) من البتات. ويتم إنشاء الإطارات عن طريق إلحاق بتات التكافؤ والمزامنة بالبيانات الخاصة بنا.




إطار تسلسلي. بعض محتويات هذا الإطار يُمكن تغيير حجم البتات الخاص بها.

دعونا نتناول بالتفصيل كل من مكونات الإطار تلك :

### كتلة البيانات (data chunk)

الجزء الحيوي والأهم في كل حزمة اتصال تسلسلي هي البيانات التي تحملها. ويُستخدم مُصطلح الكتلة لأن حجم البيانات لا يكون مُحدداً بدقة. فكمية البيانات التي تحتويها أي حزمة يُمكن أن تتراوح بين 5 و9 بتات. في الواقع الحجم القياسي للبيانات هو البايت الذي يحتوي على ثمانية بتات، لكن هناك أحجام أخرى لها استخدامات خاصة بها. فمثلاً كتلة البيانات المُكونة من سبعة بتات أفضل من المكونة من ثمانية بتات، خاصة في حالة نقل أحرف أسكي (ASCII) ذات السبعة بتات.

بعد تحديد طول الحرف يجب أن يتفق كلا الجهازين التسلسليين في ترتيب البيانات (endianness) الخاصة بهما. ترتيب البيانات يوضح كيفية نقل البيانات: هل يتم نقلها من البت الأكثر أهمية (most-significant bit (msb)) للأقل أم العكس؟ إذا لم يكن ذلك موضحاً يُمكننا الافتراض أن البيانات يتم نقلها من البت الأقل أهمية (least-significant bit (lsb)) للأكبر.

## بتات المزامنة(Synchronization bits)

بتات المزامنة هي عبارة عن بتات خاصة عددها اثنين أو ثلاثة يتم نقلها مع كل كتلة من كتل البيانات. وهي عبارة عن بت البداية (start bit) وبت(ات) التوقف (stop bit(s)). تقوم تلك البتات -كما يبدو من أسماءها – بتحديد بداية ونهاية الحزمة. دائماً يكون هناك بت بداية واحد، لكن عدد بتات التوقف يُمكن جعله واحد أو اثنين (في الغالب يتم تركه واحد) .

يتم الرمز للبداية دائماً ببت ذي حالة منخفضة (0)، بينما يُرمز للتوقف ببت ذي حالة مرتفعة (1 ) .

## بتات التكافؤ(Parity bits)

بتات التكافؤ هي عبارة عن فحص بسيط وبدائي للأخطاء. والتكافؤ له نوعان: إما فردي (odd)أو زوجي .(even) لإنشاء بت التكافؤ يتم جمع جميع البتات (بين 5-9) الموجود ببايت البيانات ومن ثم تُحدد قيمة المجموع قيمة بت التكافؤ (1 أو 0). على سبيل المثال إذا كان التكافؤ زوجي وكان بايت البيانات هو 0 b01011101والذي يحتوي على عدد فردي من القيمة 1 (خمسة)، عندها تكون قيمة بت التكافؤ 1. في المقابل إذا كان التكافؤ فردي فإن قيمة بت التكافؤ تكون في هذه الحالة 0.

استخدام التكافؤ اختياري، وفي الواقع لا يستخدم بشكل كبير. من الممكن أن يساعد في نقل البيانات خلال الأوساط ذات الضجيج(noisy mediums) ، لكن ذلك سيؤدي إلى إبطاء نقل البيانات، كما سيتطلب أن يقوم كل من المُرسل والمستقبل بمعالجة الأخطاء (في الغالب يكون لا بد من إعادة إرسال البيانات التي حدث خطأ في استقبالها).

## بروتوكول 8N1 9600 (مثال)

بروتوكول 8N1 9600 يحتوي على 8 بتات، ومعدل بود له 9600، ويحتوي على بت توقف وحيد ولا يستخدم التكافؤ. وهو من أكثر البروتوكولات التسلسلية شيوعاً. إذاً كيف تبدو حزم البيانات الخاصة ببروتوكول 8N1 9600؟ دعونا نطلع على مثال.

في حالة وجود جهاز يقوم بنقل حرفي "O" و "K" من حروف أسكي يجب إنشاء حزمتي بيانات. قيمة حرف O (كبير) في أسكي هي 79، والتي يتم تمثيلها ثنائياً في ثمانية بتات لتكون 01001111، بينما "K" يتم تمثيله ثنائياً 01001011. لم يتبقى الآن سوى إضافة بتات المزامنة.

لاحظ أننا فرضنا هنا إرسال البت الأقل في الأهمية أولاً. وفي هذه الحالة يتم إرسال كل بايت كما هو مكتوب من اليمين لليسار.



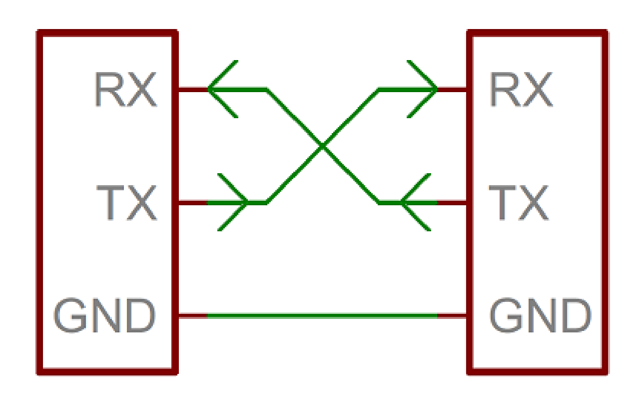

سرعة نقل البيانات في هذا البروتوكول هي bps9600 ، لذلك يكون الوقت المستغرق لجعل كل بت من هذه البتات في حالة مرتفعة أو منخفضة هو 1/9600 ثانية (104 µs ) .

كل بايت يتم إرسال البيانات من خلاله يحتوي على عشرة بتات: بت للبداية، ثمانية بتات بيانات، وبت للتوقف. لذلك وبسبب أن معدل بود 9600 فإننا يُمكننا القول إنه يتم إرسال 9600 بت أو 960 (10/9600) بايت في كل ثانية.

والآن بعد أن تعرفنا على تركيب حزم الاتصال التسلسلي يُمكننا أن ننتقل إلى الجزء الخاص بالعتاد. سنتعرف الآن كيف نحصل على قيم 1 و0 وكيف يتم احتساب معدل بود.

## التوصيلات والعتاد (Wiring and Hardware)

يتكون ناقل البيانات التسلسلي من سلكين فقط -واحد لإرسال البيانات وآخر لاستقبالها. لذلك تحتوي الأجهزة التسلسلية على منفذين تسلسليين: المُستقبل (receiver) ويُختصر RX، والمرسل (transmitter) ويُختصر TX:




عند التوصيل بين جهازين تسلسلياً يجب أن يتم توصيل منفذ RX في أحد الجهازين بمنفذ TX في الجهاز الآخر والعكس صحيح. قد تشعر بأن هذا أمر غريب لأننا سابقاً اعتدنا على توصيل المنافذ المتشابهة معاً مثل توصيل بمنفذ VCC بـ VCC وتوصيل منفذ GND بـ GND وهكذا، لكن إذا فكرت في الأمر هذه المرة فستجد أنه منطقي. فالمُرسل يجب أن يتصل بالمستقبل وليس بمرسل آخر.

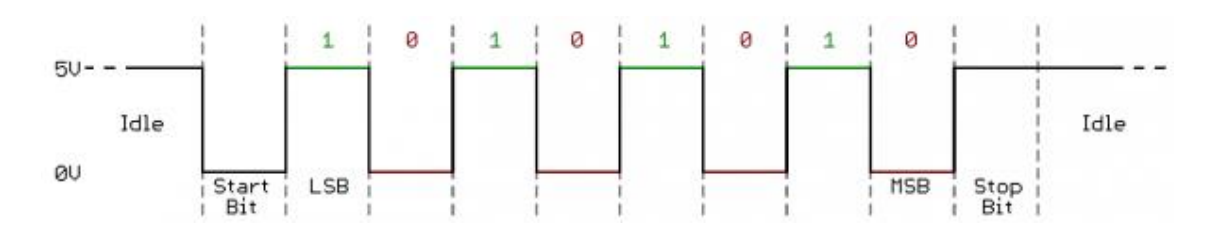
الواجهة التسلسلية التي تحتوي على جهازين يستطيع كل منهما إرسال واستقبال البيانات تكون إما ثنائية الاتجاه (**full-duplex**) أو أحادية الاتجاه (**half-duplex**). في الواجهات ثنائية الاتجاه يستطيع كلا الجهازين الإرسال والاستقبال في نفس الوقت، بينما في الواجهات أحادية الاتجاه يتناوب الجهازان في الإرسال والاستقبال ولا تتم العمليتان في ذات الوقت.

بعض النواقل التسلسلية تحتوي على سلك اتصال واحد بين جهاز مُرسل وجهاز مُستقبل. على سبيل المثال شاشات LCD التسلسلية تقوم باستقبال البيانات فقط ولا تقوم بإرسال أي بيانات إلى جهاز التحكم. وهذا ما يُطلق عليه الاتصال التسلسلي البسيط (serial communication **simplex**). وكل ما تحتاجه في هذه الحالة سلك وحيد يصل بين منفذ TX في جهاز التحكم بمنفذ RX في الجهاز المُستقبل.

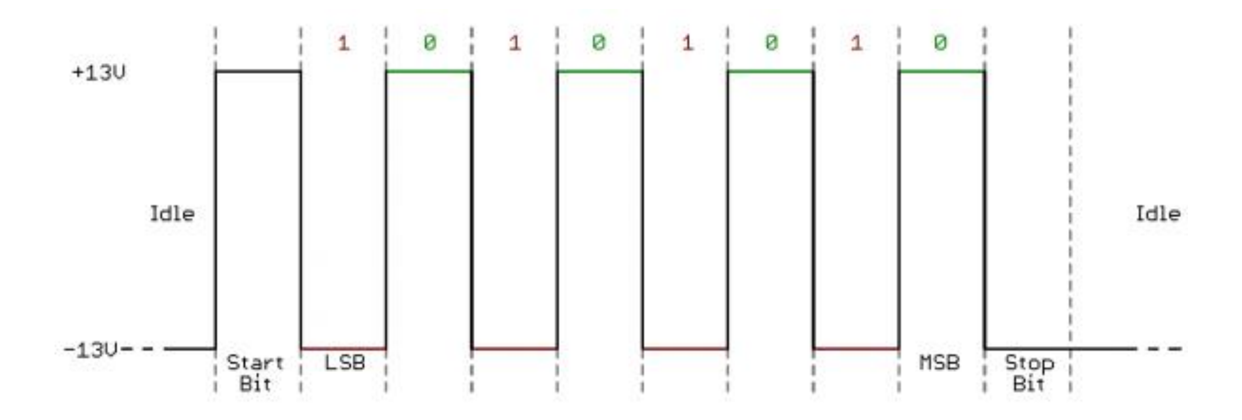
## تنفيذ العتاد(Hardware Implementation)

قمنا بتناول الاتصال التسلسلي الغير متزامن من الناحية النظرية. نحن الآن نعرف الأسلاك التي نحتاج إليها. لكن كيف يتم تنفيذ الاتصال التسلسلي على مستوى الإشارات؟ في الواقع هذا يتم بعدة طرق. هناك الكثير من معايير (standards) إرسال الإشارات التسلسلية. دعونا نتناول اثنين من أشهر تلك الواجهات: المستوى المنطقي (-logic level (TTL)) وRS-232.

عندما تتصل المُتحكمات الدقيقة والدوائر المتكامل ذات المستوى المنخفض (-low level ICs) تسلسلياً فغالباً ما تقوم بذلك على مستوى منطق الترانزستور– ترانزستور (transistor-transistor logic (TTL)). إشارات TTL التسلسلية تقع في مدى مصدر الجهد الخاص بالمًتحكم الدقيق – غالباً بين V0 وV3.3 أو V5. الإشارة التي تساوي قيمة VCC (3.3V أو V5 أو غير ذلك) تٌشير إلى خط ساكن (idle line)، أو بت بيانات قيمته 1 أو بت توقف. بينما الإشارة التي تساوي قيمتها (GND0 V) تمثل إما بت بداية أو بت بيانات قيمته 0.




واجهة RS-232 التسلسلية التي توجد في أجهزة الكمبيوتر القديمة والأجهزة الطرفية تشبه واجهة TTL مقلوبة رأساً على عقب. في الغالب تتراوح قيمة إشارات واجهة RS-232 بين -V13 و V13 بالرغم من أنها تسمح بأي قيمة بين +/- V3 و +/- V25. في هذه الإشارات يُشير الجهد المنخفض (-V5، -13V، الخ) إلى خط ساكن أو بت توقف أو بت بيانات قيمته 1. أما الإشارة المرتفعة (الموجبة) في واجهة -RS 232 فتشير إلى إما بت بداية أو بت بيانات قيمته 0. أي أن هذه الواجهة معاكسة تماماً لواجهة TTL التسلسلية.




من بين هاتين الواجهتين التسلسليتين تُعتبر واجهة TLL أسهل بكثير في التنفيذ بالدوائر الإلكترونية المُدمجة. لكن مستويات الجهد المنخفضة أكثر عرضة للفقد البيانات خلال خطوط النقل الطويلة. معيار RS-232 أو المعايير الأكثر تعقيداً مثل RS-485 تتناسب بشكل أفضل مع خطوط النقل التسلسلي الطويلة.

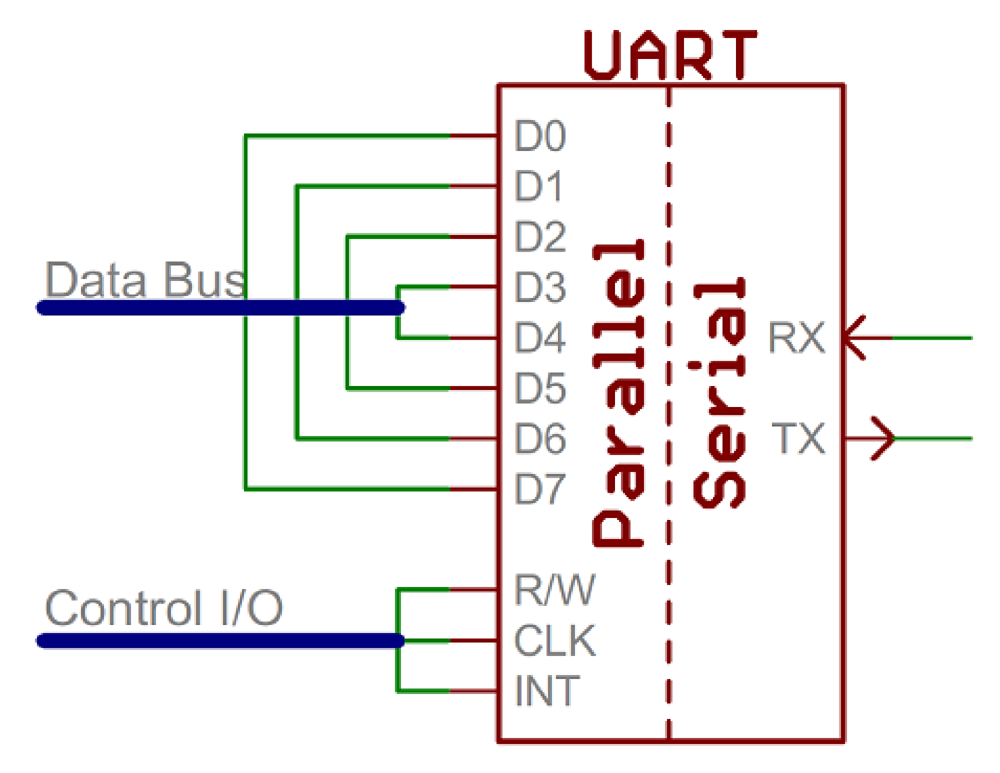
عندما تقوم بتوصيل جهازين تسلسليين معاً من المهم للغاية أن تتأكد أن جهود الإشارات الخاصة بهما متوافقة. فمثلاً لا تستطيع توصيل جهاز يعمل بواجهة TLL بجهاز آخر ذي ناقل بواجهة RS-243 مباشرة، وسيكون عليك تعديل تلك الإشارات ( shift those signals).

سنستمر في درسنا هذا وسنشرح الأداة التي تستخدمها المُتحكمات الدقيقة لتحويل بياناتها الموجودة في ناقل متوازي (parallel bus) من وإلى الواجهات التسلسلية. المرسل/المُستقبل العالمي الغير تزامني (UART).

## المرسل/المُستقبل العالمي الغير تزامني (universal asynchronous receiver/transmitter) (UART)

الجزء الأخير فيما يتعلق بالواجهات التسلسلية هو العثور على شيء يستطيع إنشاء الحزم التسلسلية والتحكم في خطوط العتاد المادي. المرسل/المُستقبل العالمي الغير تزامني (UART).

المرسل/المُستقبل العالمي الغير تزامني (UART) هو عبارة عن دائرة إلكترونية مسئولة عن تنفيذ الاتصال التسلسلي. في الأساس يعمل UART كوسيط بين الواجهات التسلسلية والمتوازية. في أحد نهايتي UART يوجد ناقل يحتوي على ثمانية خطوط بيانات أو نحو ذلك (بالإضافة لبعض منافذ التحكم (control pins))، وفي النهاية الأخرى يوجد منفذين تسلسليين RX و TX.




واجهة UART مُبسطة للغاية. لاحظ وجود ناقل متوازي في أحد الجانبين، وناقل تسلسلي في الجانب الآخر.

يُمكن أن توجد دوائر UART كدوائر متكاملة مستقلة، لكنها في الغالب توجد بداخل المتحكمات الدقيقة. يجب أن تقوم بقراءة صحيفة البيانات (datasheet) الخاصة بالمُتحكم الدقيق الذي لديك لمعرفة ما إذا كان يحتوي على UART. بعض المُتحكمات الدقيقة لا تحتوي على UART، وبعضها يحتوي على واحد، والبعض يحتوي على أكثر من ذلك. على سبيل المثال تحتوي بطاقة أردوينو أونو (Arduino Uno) المحتوية على المُتحكم الدقيق ATmega328 على UART واحد، بينما بطاقة أردوينو ميجا (Arduino Mega) المُحتوية على المتحكم الدقيق ATmega2560 تحتوي على أربعة UART.

المرسل/المُستقبل العالمي الغير تزامني (UART) – كما يبدو من اسمه - مسئول عن إرسال واستقبال البيانات التسلسلية. ففيما يتعلق بالإرسال يقوم UART بإنشاء حزم البيانات ويضيف إليها بتات المزامنة والتكافؤ، ثم يقوم بإرسال تلك الحزم عبر خط TX بتوقيت دقيق (طبقاً لمُعدل بود المُحدد). أما فيما يتعلق بالاستقبال يقوم UART باستقبال الإشارات الآتية على خط RX بمُعدل طبقاً لمُعدل بود المتوقع، مع تحديد بتات المُزامنة واستخراج البيانات.

الصورة المجاورة هي رسم تخطيطي للمكونات الداخلية لـ UART

دوائر UART الأكثر تطوراً تقوم بإدخال البيانات التي تستقبلها إلى ذاكرة وسيطة (buffer)حيث تبقى بداخلها إلى أن يلتقطها المُتحكم الدقيق. وتقوم دوائر UART بإخراج البيانات التي تختزنها بمبدأ من يأتي أولاً يخرج أولاً (first-in-first-out (FIFO))من المُمكن أن تكون الذاكرة الوسيطة صغيرة للغاية لا تتعدى بضعة بتات، ومن الممكن أن تكون كبيرة لتبلغ آلاف البايتات.

## المرسل/المُستقبل العالمي الغير تزامني البرمجي(Software UART)

إذا لم يكن المُتحكم الدقيق يحتوي على UART (أو كان يحتوي على عدد أقل من المطلوب) يُمكن أن يتم استخدام تقسيم البت (Bit-banging) للواجهة التسلسلية، والذي يتم التحكم به بواسطة المُعالج (processor) مباشرة. وهذا هو الأسلوب الذي تتبعه مكتبات أردوينو مثل SoftwareSerial. تقسيم البت يستهلك المُعالج وفي الغالب لا تصل دقته لـ UART، لكنه يفي بالغرض إلى حد ما.

## أخطاء شائعة

تحدثنا عن كل ما يتعلق بالاتصال التسلسلي. أود أن أترككم مع بعض الأخطاء الشائعة التي يُحتمل أن يقع فيها أي شخص مُهتم بالإلكترونيات مهما بلغت خبرته :

## توصيل RX بـ TX و TXبـRX

يبدو أمراً بسيطاً، لكنه خطأ سهل الوقوع فيه. تأكد دائماً من توصيل خط RX بخط TXبين الأجهزة التسلسلية.



بطاقة .Pro Mini لاحظ توصيل منفذ RX بـ.TX

## عدم تطابق مُعدل بود

مُعدل بود يشبه اللغة التي تتحدث بها الاتصالات التسلسلية. إذا لم يكن الجهازان يتحدثان بنفس اللغة، فإما أن تحدث أخطاء في تفسير البيانات أو تُفقد بشكل تام. تأكد دائماً من أن مُعدل بود للأجهزة التسلسلية التي تقوم بتوصيلها معاً متماثل.

COM29:19200baud - Tera Term VT

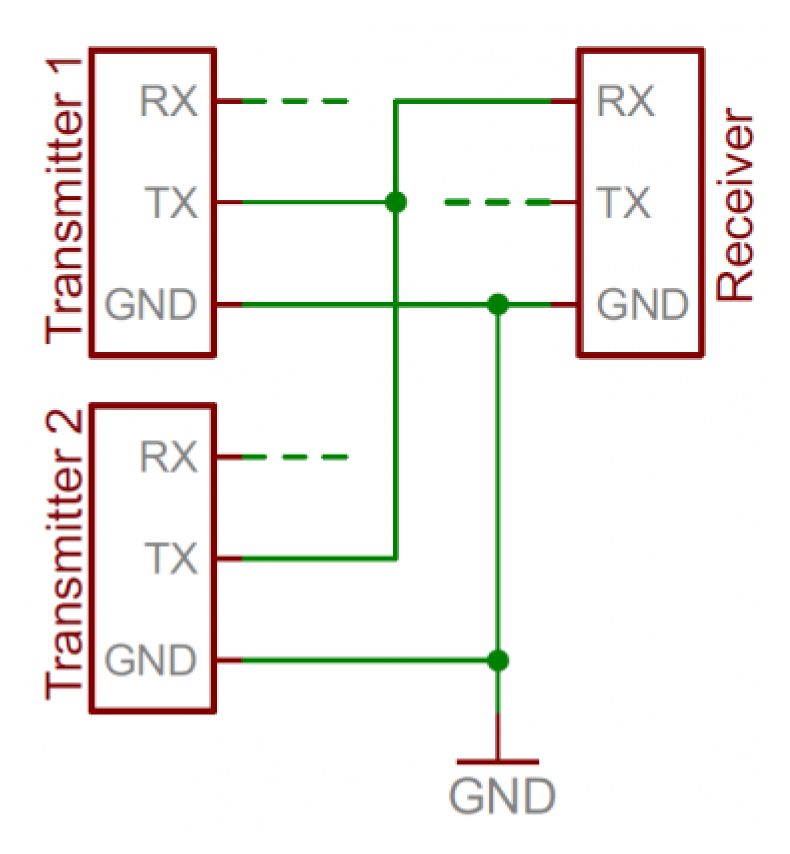
File Edit Setup Control Window Help

بيانات يتم إرسالها بمعدل bps9600، ولكن يتم استقبالها بمُعدل 19200 .bpsعدم تطابق معدل بود = بيانات مفقودة

## منافسة الناقل(Bus Contention)

تم تصميم الاتصال التسلسلي للسماح لجهازين فقط بالاتصال عبر ناقل تسلسلي واحد. وإذا حاول أكثر من جهاز واحد الإرسال على نفس الخط التسلسلي يُمكن أن يتسبب ذلك فيما يُعرف بمنافسة الناقل(bus-contention) .

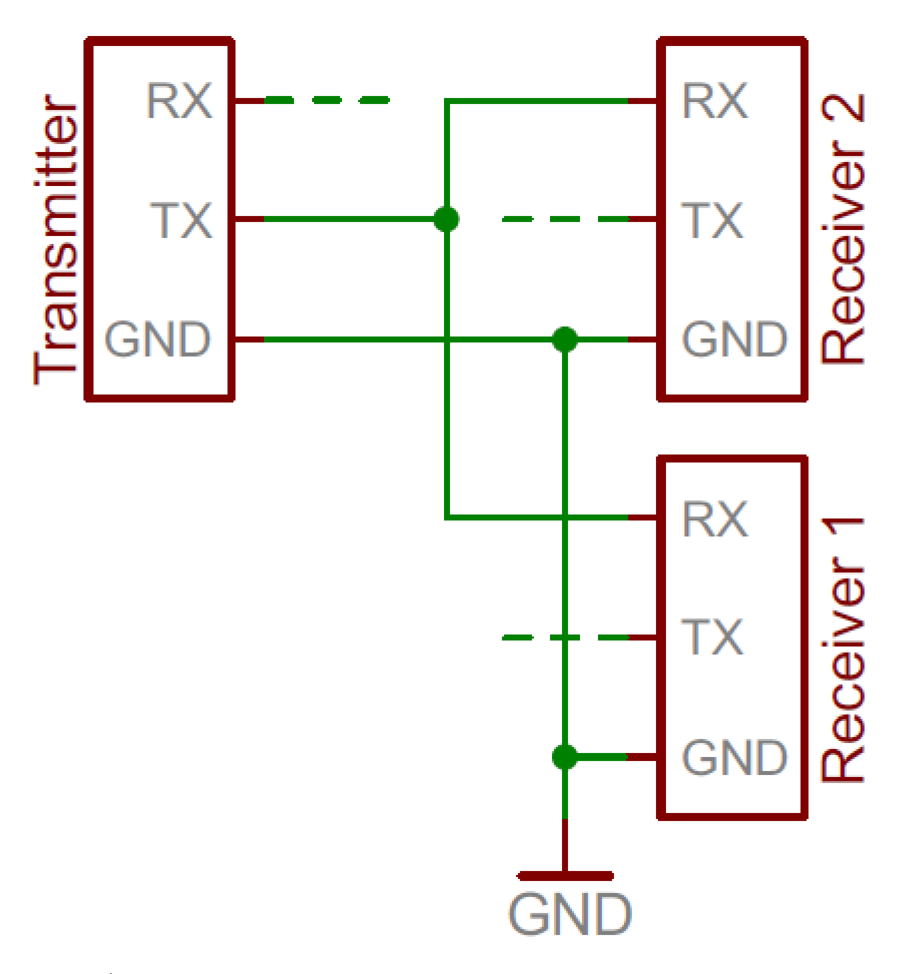
على سبيل المثال إذا أردت بتوصيل وحدة GPS ببطاقة أردوينو فستقوم بتوصيل خط TXالخاص بالوحدة بخط RX على بطاقة الأردوينو. لكن منفذ RX الخاص ببطاقة الأردوينو مُتصلة بالفعل من قبل بمنفذ TX الخاص بالمحول من USB إلى تسلسلي (USB-to-serial converter)الذي يُستخدم عند الحاجة لبرمجة بطاقة الأردوينو أو استخدام الشاشة التسلسلية .(Serial Monitor) في هذه الحالة قد يحدث أن تُحاول كل من وحدة GPS ورقاقة FTDI الإرسال على نفس الخط في نفس الوقت.




مُرسلان يقومان بالإرسال إلى مُستقبل واحد، مما يُسبب احتمالية حدوث منافسة الناقل

وجود جهازين يُحاولان إرسال البيانات في نفس الوقت على نفس الخط أمر سيء. ففي أفضل الحالات لن يتمكن أي من الجهازين من إرسال البيانات، وفي أسوأ الحالات يتلف خطا الإرسال لكلا الجهازين (لكن هذا نادر الحدوث).

على الجانب الآخر يُمكن توصيل جهازي استقبال بخط إرسال واحد بأمان. هذا ليس أمر شائع ولكنه يعمل. على سبيل المثال إذا أردت توصيل شاشة LCD تسلسلية ببطاقة أردوينو فأسهل شيء تقوم به هو توصيل خط RX الخاص بوحدة LCD بخط TX الخاص ببطاقة الأردينو، وفي الوقت ذاته يكون منفذ TX الخاص بباقة الأردوينو مُتصلاً بخط RX الخاص بمُبرمج USB، ولكن في هذه الحالة يبقى هناك جهاز واحد يقوم بالإرسال.




لكن توصيل خط TX بهذا الشكل يبقى من المُمكن أن يُشكل خطراً على البرنامج (firmware)، لأنك لن تتمكن من تحديد أي جهاز يستقبل أي إشارات يتم إرسالها. فشاشة LCD قد تستقبل بيانات ليست مُهيأة لاستقبالها والتعامل معها، والذي يمكن أن يتسبب في دخولها في حالة غير معروفة.

بوجه عام يُفضل استخدام ناقل تسلسلي واحد لكل جهازين تسلسليين.

# واجهة الاتصال التسلسلية في أردوينو

تستخدم واجهة التخاطب التسلسلية للتواصل بين لوحة أردوينو وأجهزة أخرى (مثل الحاسوب). تملك جميع لوحات أردوينو منفذ تسلسلي واحد على الأقل يُعرَف باسم UARTأو USART يستعمل هذا المنفذ الرجل 0 (RX) والرجل 1 (TX) في عملية التواصل، إذ تكون هاتان الرجلان موصلتين مع المنفذ USB للتواصل مع الحاسوب. نتيجةً لذلك، إن استعملت إحدى الدوال الموجودة في هذا القسم في تهيئة واجهة التخاطب التسلسلية والبدء باستعمالها، فلن تتمكن من استعمال الرجلين 0 و 1 من أجل الدخل أو الخرج الرقمي بعدئذٍ. تستطيع استعمال مراقب الاتصالات التسلسلية المدمج في بيئة أردوينو التطويرية (Arduino IDE) للتواصل مع لوحة أردوينو. اضغط على زر مراقب الاتصال التسلسلي (serial monitor) في شريط الأدوات واختر معدل تدفق البيانات أو سرعة الاتصال (baud rate) نفسه الذي استُعمَل في تهيئة الاتصال باستدعاء التابع begin().

يستعمل الاتصال التسلسلي في الرجلين RX و TX المستويات TTL المنطقية (5 V أو 3.3 Vبناءً على اللوحة المستعملة). لا توصل هذه الأرجل مباشرةً مع منفذ RS232تسلسلي، إذ يعمل بجهد 12 +/- Vوقد يتسبِّب في إتلاف اللوحة.

تملك لوحة أردوينو Mega ثلاثة منافذ اتصال تسلسلية إضافية هي: المنفذ Serial1 على الرجلين 19 (RX) و 18(TX) ، والمنفذ Serial2 على الرجلين 17 (RX) و 16(TX) ، والمنفذ Serial3 على الرجلين 15 (RX) و 14 (TX).إن أردت استعمال هذه الأرجل في التواصل مع الحاسوب، فستحتاج إلى محول من المنفذ USB إلى منفذ تسلسلي .(USB-to-serial adaptor) أمّا إن أردت استعمال هذه الأرجل للتواصل مع جهاز TTL تسلسلي خارجي، فصِل الرجل TX للوحتك مع الرجل RX لذلك الجهاز، والرجل RX للوحتك مع الرجل TX للجهاز الخارجي، والقطب الأرضي للوحتك (Mega) مع القطب الأرضي للجهاز.

تملك لوحة أردوينو Due ثلاثة منافذ TTL تسلسلي بجهد 3.3 V إضافية هي: المنفذ Serial1 على الرجلين 19 (RX) و 18(TX) ، والمنفذ Serial2 على الرجلين 17 (RX) و 16(TX) ، والمنفذ Serial3 على الرجلين 15 (RX) و 14 (TX).الرجلان 0 و 1 موصولتان أيضًا مع ما يقابلهما في المتحكم ATmega16U2المستعمل كشريحة تحويل من منفذ USB إلى منفذ TTL تسلسلي،

والموصول أيضًا مع المنفذ USB لتنقيح الأخطاء .(USB debug port) أضف إلى ذلك أنه يوجد منفذ USB تسلسلي أصلي على شريحة المتحكم SAM3X يدعى SerialUSB.

تستعمل لوحة أردوينو Leonardo المنفذ Serial1 التسلسلي للتواصل عبر المنفذ TTL بجهد 5 V في الرجلين 0 (RX) و 1 (TX) المنفذ Serial محجوزٌ للاتصال USB CDC

**الاوامر المستخدمة :**

Serial.if(Serial)

يتحقَّق التابع if(Serial) إن كان المنفذ Serial التسلسلي المُمرَّر إليه جاهزًا للاستعمال. في لوحات أردوينو Leonardo ، يتحقَّق التابع if(Serial) إن كان منفذ الاتصال USB CDC مفتوحًا أم لا. في جميع النسخ الأخرى (من ضمنها if(Serial1)) في لوحات Leonardo ، ذلك الاستدعاء سيعيد دومًا القيمة true.

الصيغة العامة :

```
// جميع اللوحات
if (Serial)

// لوحات أردوينو Leonardo
if (Serial1)

// لوحات أردوينو Mega
if (Serial1)
if (Serial2)
if (Serial3)
```

تعاد القيمة true المنطقية إن كان منفذ الاتصال التسلسلي المحدَّد متاحًا. ستعاد القيمة falseإن استُعلِم عن الاتصال USB CDC التسلسلي في لوحات Leonardo قبل أن يكون جاهزًا.

مثال على تهيئة الاتصال التسلسلي وانتظار فتح المنفذ:

```
void setup() {
 // تهيئة الاتصال التسلسلي والانتظار اكتمال فتح المنفذ
  Serial.begin(9600);
  while (!Serial) {
    ; // انتظار منفذ الاتصال التسلسلي لكي يتصل. هذا الأمر
الأصلي USB مطلوب من أجل المنفذ
  }
}

void loop() {
 // تنفيذ بقية المهام بشكل طبيعي
}
```

## Serial.available()

يجلب التابع available() عدد البايتات (المحارف) المتاحة للقراءة من منفذ الاتصال التسلسلي. يمثِّل هذا العدد كمية البيانات التي استُقبِلَت مسبقًا وخُزِّنَت في ذاكرة التخزين المؤقتة للمنفذ التسلسلي ذي الحجم 64 بايت.

الصيغة العامة :

```
Serial.available()

// لوحات أردوينو Mega فقط
Serial1.available()
Serial2.available()
Serial3.available()
```

المثال التالي خاص بلوحة أردوينو Mega. يعمل المثال على إرسال البيانات المستلمة من أحد منافذ الاتصال التسلسلية إلى منفذ آخر. يمكن تطبيق هذا الأمر لوصل جهاز مع الحاسوب مثلًا عبر لوحة أردوينو:

```
void setup() {
  Serial.begin(9600);
  Serial1.begin(9600);

}

void loop() {
  // القراءة من المنفذ 0، والإرسال إلى المنفذ 1
  if (Serial.available()) {
    int inByte = Serial.read();
    Serial1.print(inByte, DEC);

  }
  // القراءة من المنفذ 1، والإرسال إلى المنفذ 0
  if (Serial1.available()) {
    int inByte = Serial1.read();
    Serial.print(inByte, DEC);
  }
}
```

## Serial.begin()

يضبط التابع `begin()` معدل تدفق البتات في الثانية الواحدة (baud) لبدء عملية نقل البيانات عبر الاتصال التسلسلي. من أجل التواصل مع الحاسوب، استعمل أحد معدلات تدفق البيانات التالية: 300، أو 600، أو 1200، أو 2400، أو 4800، أو 9600، أو 14400، أو 19200، أو 28800، أو 38400، أو 57600، أو 115200. مع ذلك، تستطيع تحديد معدلات نقل أخرى للتواصل عبر الرجلين 0 و 1 مع جهاز أو عنصر يتطلب معدل تدفق بيانات محدَّد غير تلك المذكورة.

يضبط المعامل الثاني الاختيار المُمرَّر إلى التابع `begin()` بتات البيانات، التماثل (parity)، والإيقاف. القيمة الافتراضية هي 8 بتات للبيانات وبت للإيقاف وعدم وجود بتات للتحقق من التماثل.

الصيغة العامة :

```
Serial.begin(speed)
Serial.begin(speed, config)

// لوحات أردوينو Mega فقط
Serial1.begin(speed)
Serial2.begin(speed)
Serial3.begin(speed)
Serial1.begin(speed, config)
Serial2.begin(speed, config)
Serial3.begin(speed, config)
```

## Serial.write()

يكتب التابع write() بيانات ثنائية على منفذ الاتصال التسلسلي. تُرسَل هذه البيانات كبايت أو سلسلة من البايتات. إن أردت إرسال محارف تمثِّل أرقام عددٍ، فاستعمل التابع print()عوضًا ذلك.

الصيغة العامة

```
Serial.write(val)
Serial.write(str)
Serial.write(buf, len)

// أيضًا Serial3, Serial2, Serial1 المنافذ استعمال Mega
تدعم لوحات أردوينو
```

المعاملات :

val

قيمةٌ يراد إرسالها على أنَّها بايتٌ مفردٌ.

str

سلسلةٌ نصيةٌ يراد إرسالها بشكل سلسلة من البايتات المفردة.

buf

مصفوفةٌ يراد إرسالها بشكل سلسلة من البايتات المفردة.

استعمال التابع write() لإرسال بايت واحد وسلسلة من البايتات إلى المنفذ التسلسلي:

```
void setup(){
  Serial.begin(9600);
}

void loop(){
```

```
  Serial.write(45); // 45 إرسال بايت قيمته

   int bytesSent = Serial.write("hello"); // وتخزين طولها
"hello" إرسال السلسلة النصية بعد نجاح العملية
}
```

## Serial.print()

يطبع التابع `print()` البيانات المُمرَّرة إليه على منفذ الاتصال التسلسلي بترميز ASCII أي يطبع نصًّا يستطيع الآخرون قراءته .

تُطبَع الأعداد باستعمال الترميز ASCII لكل رقم، وتُطبَع الأعداد العشرية بشكل مشابه بترميز ASCII وبدقة عددين بعد الفاصلة افتراضيًّا. تُرسَل البايتات فرادى، كل محرف على حدة، وتُرسَل المحارف والسلاسل النصية كما هي.

```
Serial.print(78)             // "78" يعطي القيمة
Serial.print(1.23456)        // "1.23" يعطي القيمة
Serial.print('N')            // "N" يعطي القيمة
Serial.print("Hello world.") // "Hello world" يعطي نفسها
السلسلة
```

يمكن تمرير معاملٍ ثانٍ اختياريٍّ إلى التابع `print()` يحدِّد التنسيق (الأساس ([base]) المراد استعماله لطباعة البيانات. أمَّا من أجل الأعداد العشرية، فيحدِّد هذا المعامل عدد الأرقام بعد الفاصلة.

```
Serial.print(78, BIN)     // "1001110" يعطي القيمة
Serial.print(78, OCT)     // "116" يعطي القيمة
Serial.print(78, DEC)     // "78" يعطي القيمة
Serial.print(78, HEX)     // "4E" يعطي القيمة
Serial.print(1.23456, 0) // "1" يعطي القيمة
Serial.print(1.23456, 2) // "1.23" يعطي القيمة
Serial.print(1.23456, 4) // "1.2346" يعطي القيمة
```

تستطيع تمرير سلاسل نصية مخزَّنة في ذاكرة البرنامج (flash-memory) إلى التابع print() عبر تغليفها بالتابع F()

```
Serial.print(F("Hello World"))
```

إن أردت إرسال البيانات دون إجراء أية عملية تحويل عليها، فاستعمل التابع write().

اذا الصيغة العامة هي :

```
Serial.print(val)
Serial.print(val, format)
```

## Serial.println()

يطبع التابع println() البيانات المُمرَّرة إليه على منفذ الاتصال التسلسلي بترميز ASCII (أي نص يستطيع الأشخاص الآخرين قراءته) ثمَّ يُتبعها بمحرف العودة إلى بداية السطر (المحرف '\r' أو ASCII 13) ومحرف سطر جديد (المحرف '\n' أو ASCII 10). بعبارة أخرى، يشبه هذا التابع التابع print() تمامًا باستثناء أنه يضيف محرف العودة إلى بداية السطر ومحرف سطر جديد إلى نهاية البيانات التي يطبعها، لذا ارجع إلى التابع print() لتفاصيل وشرح أوسع.

الصيغة العامة

```
Serial.println(val)
Serial.println(val, format)
```

قراءة دخل تشابهي من الرجل 0 وإرسال القيمة عبر منفذ الاتصال التسلسلي:

```
int analogValue = 0;    // متغير لتخزين القيمة المراد قراءتها

void setup() {
  // فتح منفذ اتصال تسلسلي بمعدل 9600 بت/ثا
  Serial.begin(9600);
}

void loop() {
  // قراءة الدخل التشابهي من الرجل 0
  analogValue = analogRead(0);

  // طباعة القيمة المقروءة بتنسيقات مختلفة
  Serial.println(analogValue);       // print as an ASCII-encoded decimal
  Serial.println(analogValue, DEC);  // print as an ASCII-encoded decimal
  Serial.println(analogValue, HEX);  // print as an ASCII-encoded hexadecimal
  Serial.println(analogValue, OCT);  // print as an ASCII-encoded octal
  Serial.println(analogValue, BIN);  // print as an ASCII-encoded binary

  // الانتظار لمدة 10 ميلي ثانية قبل إعادة تكرار العملية
  delay(10);
```

# Serial.read()

يقرأ التابع `read()` البيانات المستلمة عبر الاتصال التسلسلي , والصيغة العامة هي:

```
Serial.read()

// لوحات أردوينو Mega فقط
Serial1.read()
Serial2.read()
Serial3.read()
```

قراءة البيانات المستلمة من منفذ الاتصال التسلسلي وإعادة إرسالها:

```
int incomingByte = 0;   // متغير لتخزين البيانات المراد
قراءتها

void setup() {
        Serial.begin(9600);     // فتح منفذ اتصال تسلسلي
بمعدل 9600 بت/ثا
}

void loop() {

        // إرسال بيانات عند استلام أخرى فقط
        if (Serial.available() > 0) {
                // قراءة بايتٍ من البيانات المستلمة
                incomingByte = Serial.read();

                // إعادة إرسال نفس البايت المستلم
                Serial.print("I received: ");
                Serial.println(incomingByte, DEC);
        }
}
```

يقرأ التابع readBytes() البيانات المستلمة عبر الاتصال التسلسلي ويضعها في متغير يمثِّل مخزنًا مؤقتًا (buffer). سيُنهَى عمل التابع readBytes() إن قُرِئت البايتات المحدِّدة أو انتهت المهلة الزمنية اطلع على التابع setTimeout()

```
Serial.readBytes(buffer, length)
```

يقرأ التابع readString() المحارف المستلمة من الاتصال التسلسلي ويعيدها كسلسلة نصية. سيُنهَى عمل التابع readString() إن انتهت المهلة الزمنية اطلع على التابع setTimeout()

```
Serial.readString()
```

يعطِّل التابع end() الاتصال التسلسلي المفتوح محرِّرةً بذلك الرجل 0 (RX) والرجل 1 (TX) ليعاد استعمالهما كدخل أو خرج رقمي. لإعادة تفعيل الاتصال التسلسلي، استدعي التابع begin() مجدَّدًا.

```
Serial.end()

Serial1.end()
Serial2.end()
Serial3.end()
```

يُستدعَى التابع serialEvent() عندما يكون هنالك بيانات متاحة للقراءة عبر منفذ الاتصال التسلسلي. استعمل التابع read() ضمن هذه التابع من أجل التقاط تلك البيانات وقراءتها , الصيغة العامة :

```
void serialEvent(){
    // المهام المراد تنفيذها عند استدعاء الدالة
}

// في لوحات أردوينو Mega فقط:
void serialEvent1(){
    // المهام المراد تنفيذها عند استدعاء الدالة
}

void serialEvent2(){
    // المهام المراد تنفيذها عند استدعاء الدالة
}

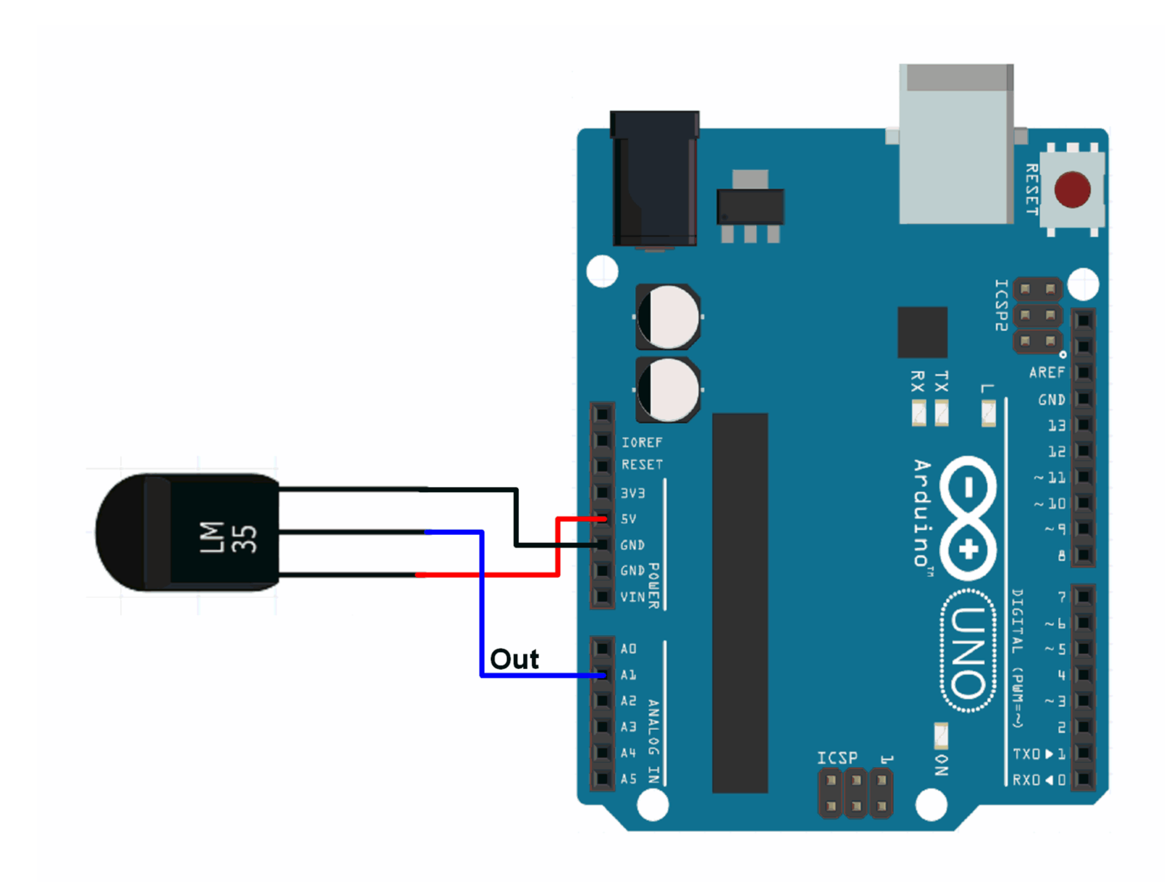
void serialEvent3(){
    // المهام المراد تنفيذها عند استدعاء الدالة
}
```

يضبط التابع setTimeout() المهلة الزمنية القصوى بالميلي ثانية لانتظار وصول البيانات من منفذ الاتصال التسلسلي. يُستعمَل هذه التابع مع التوابع serial.readBytesUntil()، أو serial.readBytes() أو serial.readString(). القيمة الافتراضية للمهلة الزمنية هي 1000 ميلي ثانية.

ويكتب بالصيغة التالية :

```
Serial.setTimeout(time)
```

- استخدام حساس درجة الحرارة LM35 لعرض درجات الحرارة , قم بتوصيل الحساس بالشكل التالي :




وادخل الكود

```
const int lm35_pin = A1;      /* LM35 O/P pin */

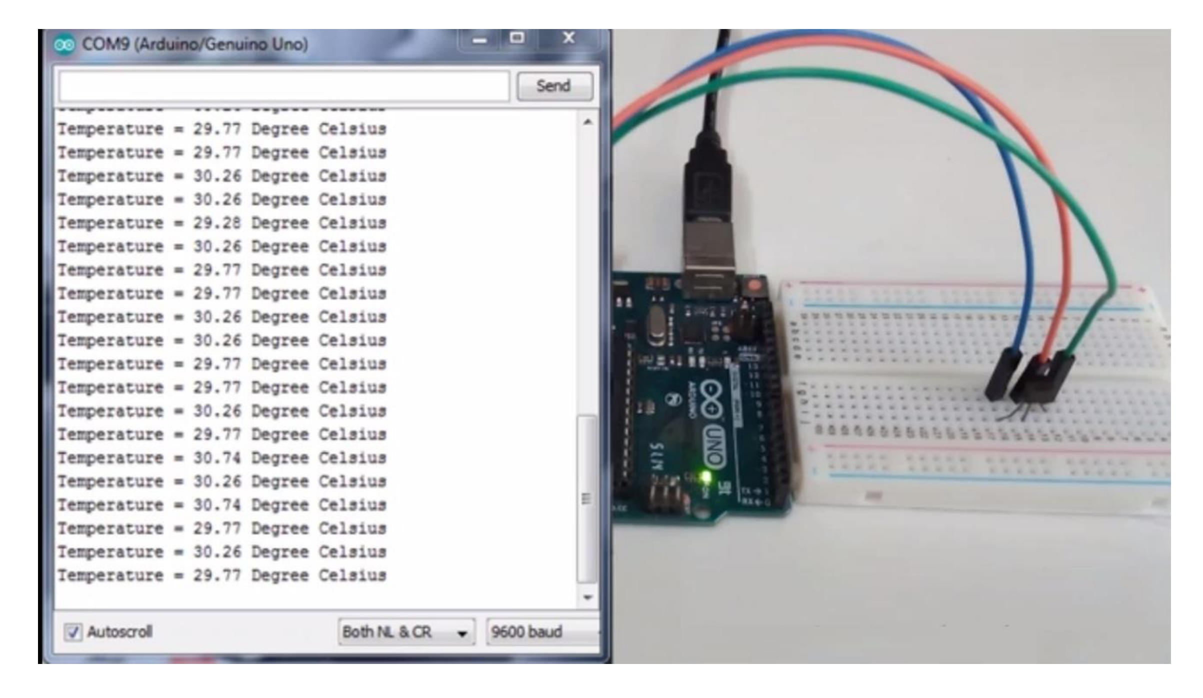
void setup() {
  Serial.begin(9600);
}
```

```
void loop() {
  int temp_adc_val;
  float temp_val;
  temp_adc_val = analogRead(lm35_pin);      /* Read Temperature */
  temp_val = (temp_adc_val * 4.88); /* Convert adc value to
equivalent voltage */
  temp_val = (temp_val/10);  /* LM35 gives output of 10mv/°C */
  Serial.print("Temperature = ");
  Serial.print(temp_val);
  Serial.print(" Degree Celsius\n");
  delay(1000);
}
```

ثم اضغط على serial Monitor

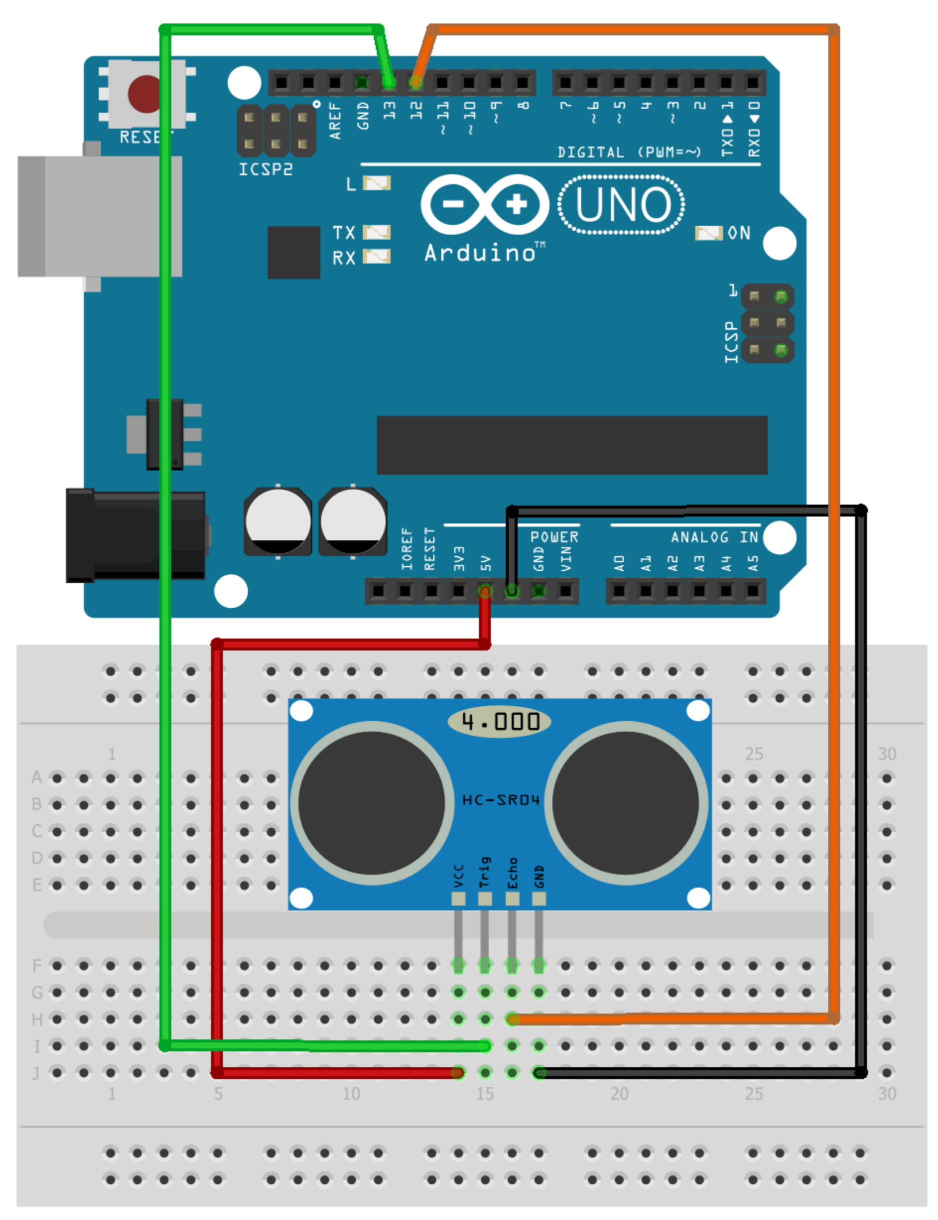



وستلاحظ ان البيانات بدئت تظهر وتتغير كل ثانية



- استخدام حساس الامواج فوق الصوتية لقياس المسافة وعرضها على الحاسوب

قم بتوصيل الحساس بالشكل التالي :



| Ultrasonic Sensor HC-SR04 | Arduino |
|---|---|
| VCC | 5V |
| Trig | Pin 13 |
| Echo | Pin 12 |
| GND | GND |

وادخل الكود

```
#define trigPin 13
#define echoPin 12

void setup() {
  Serial.begin (9600); // Begin the Serial Monitor at 9600 Baud
  pinMode(trigPin, OUTPUT); // Initialize Trigger Pin
  pinMode(echoPin, INPUT); // Initialize Echo Pin
}

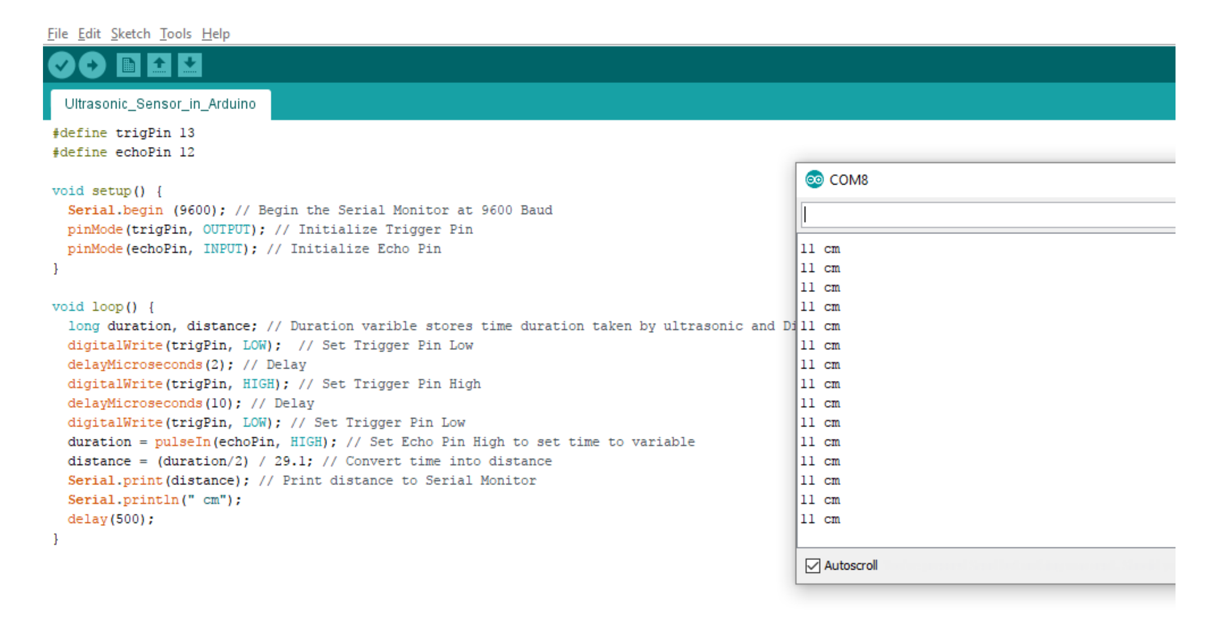
void loop() {
  long duration, distance; // Duration varible stores time duration
taken by ultrasonic and Distance variable stores distance
  digitalWrite(trigPin, LOW);  // Set Trigger Pin Low
  delayMicroseconds(2); // Delay
  digitalWrite(trigPin, HIGH); // Set Trigger Pin High
  delayMicroseconds(10); // Delay
  digitalWrite(trigPin, LOW); // Set Trigger Pin Low
  duration = pulseIn(echoPin, HIGH); // Set Echo Pin High to set
time to variable
  distance = (duration/2) / 29.1; // Convert time into distance
  Serial.print(distance); // Print distance to Serial Monitor
```

```
  Serial.println(" cm");
  delay(500);
}
```

ثم اضغط على serial Monitor




لاحظ عن تقريب وابعاد الاجسام سوف تتغير قراءات الحساس

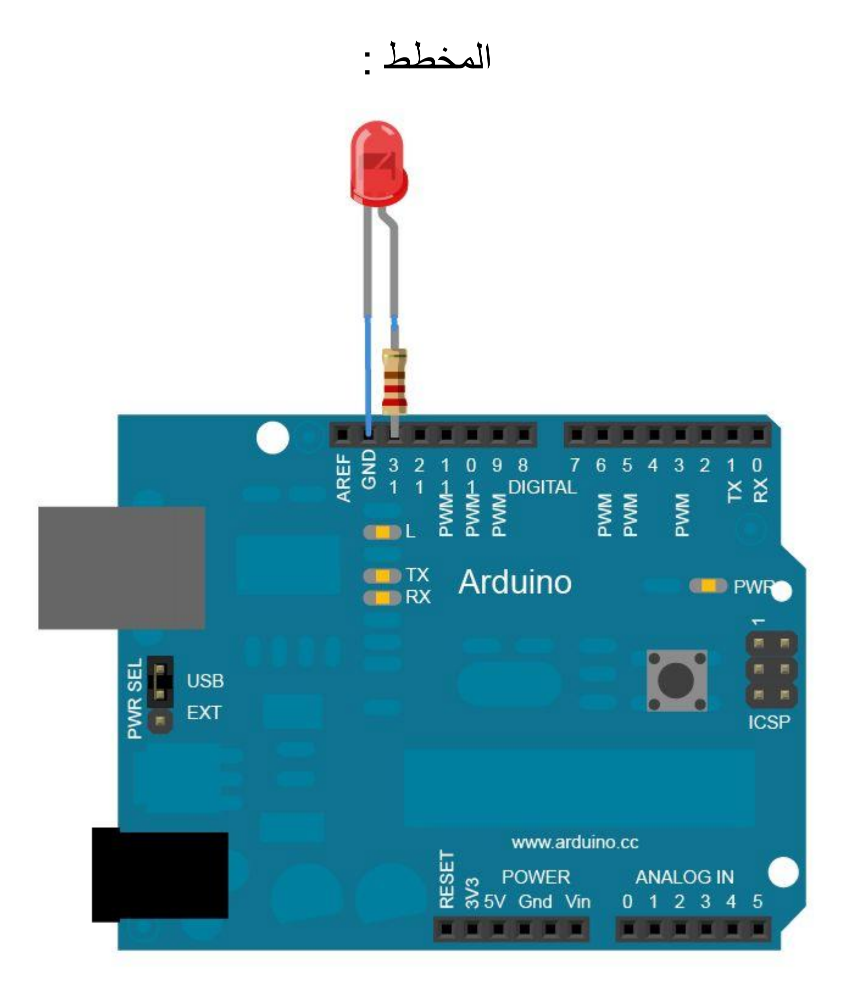


## ارسال المعلومات بواسطة serial monitor

ان نافذة serial monitor ليست فقط لاظهار النتائج بل يمكن ارسال المعلومات بواسطتها وسنبين ذلك في الامثلة .

تشغيل المصباح عند ادخال 1 واطفائه عند ادخال 2

المخطط :



الكود :

```
void setup()

{

pinMode(13, OUTPUT);

Serial.begin(9600);

while (!Serial);

Serial.println("Input 1 to Turn LED on and 2 to off");

}

void loop() {

if (Serial.available())

{

int state = Serial.parseInt();

if (state == 1)

{

digitalWrite(13, HIGH);

Serial.println("Command completed LED turned ON");

}

if (state == 2)

{
```

```
digitalWrite(13, LOW);

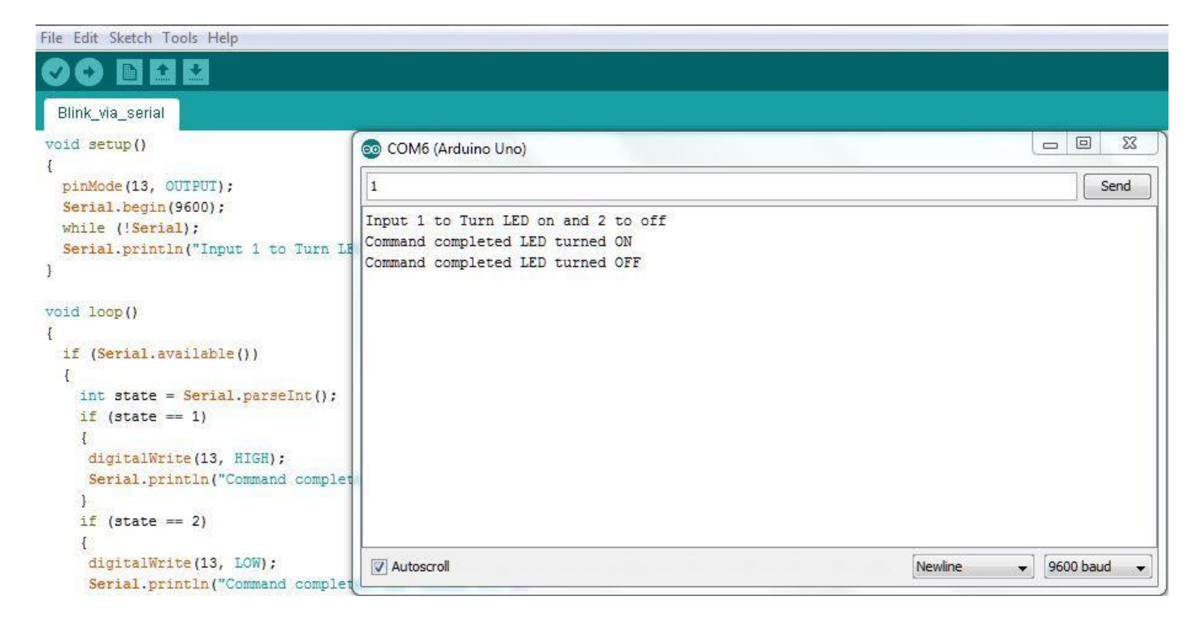
Serial.println("Command completed LED turned OFF");

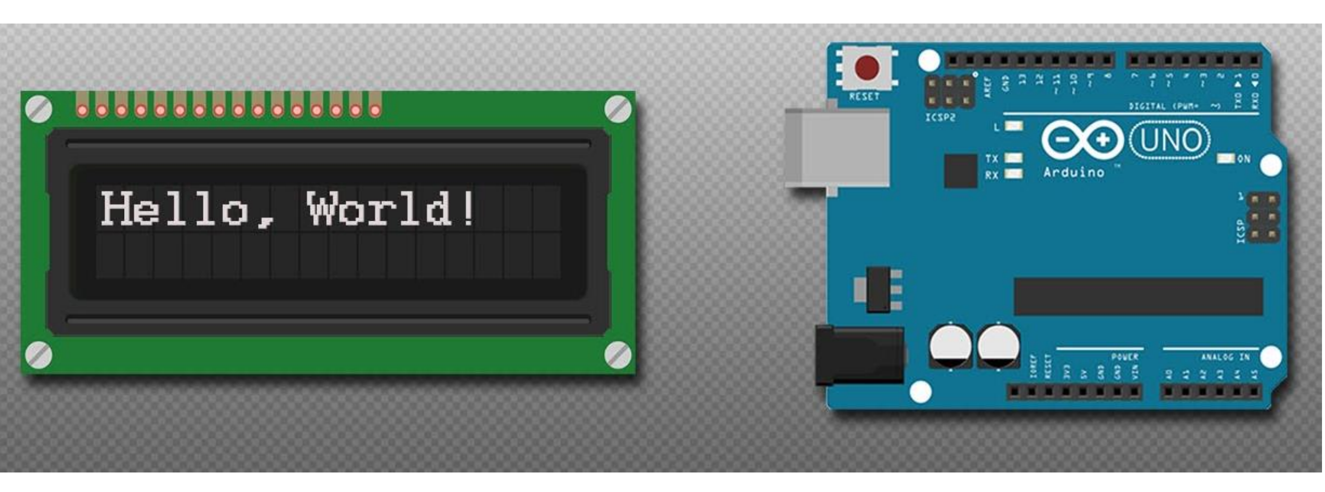
}

}

}
```

والان يمكن ادخال الاوامر باستخدام لوحة المفاتيح

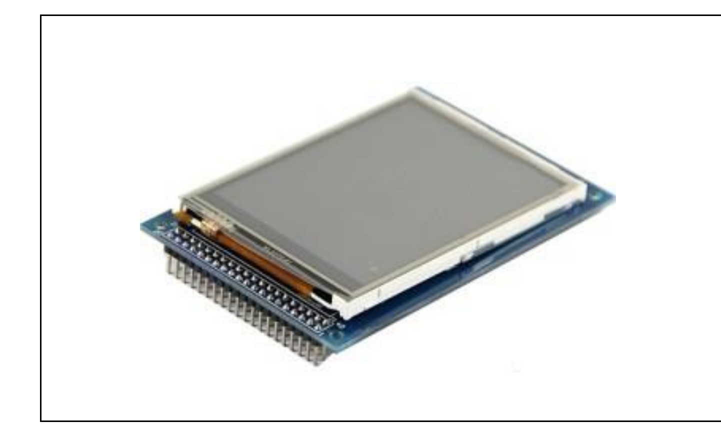





يحتوي لوح Arduino على مجموعة متنوعة من الشاشات المتوافقة التي يمكنك استخدامها في مشاريعك الإلكترونية. في معظم المشاريع ، من المفيد جدًا إعطاء المستخدم بعض التعليقات من.Arduino

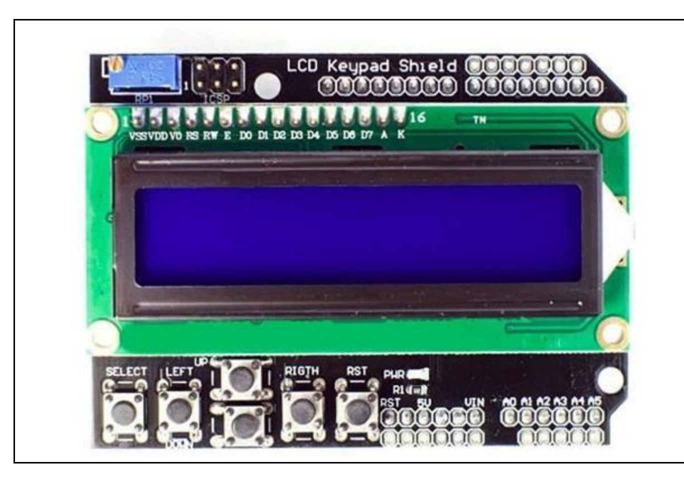
سواء كان ذلك بمثابة قراءة للمستشعر أو رسالة جيدة أو لإنشاء واجهة للتفاعل مع لوحة.Arduino

في ما يلي قائمة تضم بعض انواع الشاشات المختلفة التي يدعمها معظم انواع الاردوينو:

| | |
|---|---|
| TFT LCD display<br>يمكنك عرض الصور الملونة أو الرسومات. هذه الوحدة لديها دقة 480 × 320. هذه الوحدة تتضمن مقبس بطاقة SD و دائرة.SPI FLASH |  |

| | |
|---|---|
|  | TFT LCD touchscreen display<br><br>مشابهة للشاشة اعلاه ولكنها تعمل باللمس |
|  | White OLED display<br>هذه شاشة صغيرة بحجم 1 × 0.96 بوصة فقط. يحتوي هذا العرض على خلفية سوداء ويعرض الأحرف باللون الأبيض. هناك شاشات أخرى مشابهة يمكنها إظهار الأحرف في ألوان أخرى. |
|  | 16×2 character LCD display<br><br>عادة ما يكون هذا أول عرض يستخدمه معظم الأشخاص عند بدء استخدام لوحة Arduino.<br><br>يعرض 16 حرفًا في صفين (هناك أيضًا أحجام أخرى متوفرة). هذه الشاشات تأتي مع خلفية زرقاء أو خضراء ومع خلفية |

| | |
|---|---|
| 5110 LCD display<br><br>هذه هي أنواع الشاشات المستخدمة في الهواتف الخلوية القديمة من نوكيا.<br><br>الخلفية رمادية ويتم عرض الأحرف أو الصور بلون رمادي غامق. فهي رخيصة جدا وسهلة الاستخدام. |  |
| 4 Bits Digital Tube LED display<br><br>تسمح لك هذه الشاشة بعرض 4 أرقام مع سبعة أجزاء. من المفيد عرض البيانات من مستشعر درجة الحرارة ، على سبيل المثال. |  |
| Dot matrix<br>مصفوفة النقاط لديها 64 المصابيح ، 8 × 8. يمكنك التحكم في كل منها بشكل فردي لعرض الأحرف والأرقام والأرقام ، إلخ.<br><br>يمكنك إرفاق عدة مصفوفات نقطية ، للحصول على مساحة أكبر. |  |

| | |
|---|---|
| <br> | ARDUINO LCD + KEYPAD SHIELD<br><br>هذه الشاشة مزودة بلوحة مفاتيح خاصة |

المثال الاول وهو طباعة ("Hello World!") على الشاشة :

| القطع المطلوبة | |
|---|---|
| ohm resistor 220 | LCD Screen (compatible with Hitachi HD44780 (driver |
| pin headers | k ohm potentiometer10 |
| Wires | Arduino uno |
| Bread board | Power tool |

تسمح لك مكتبة LiquidCrystal بالتحكم في شاشات LCD المتوافقة مع برنامج Hitachi HD44780.هناك العديد منهم هناك ، ويمكنك عادة إخبارهم بواسطة واجهة 16.pin-

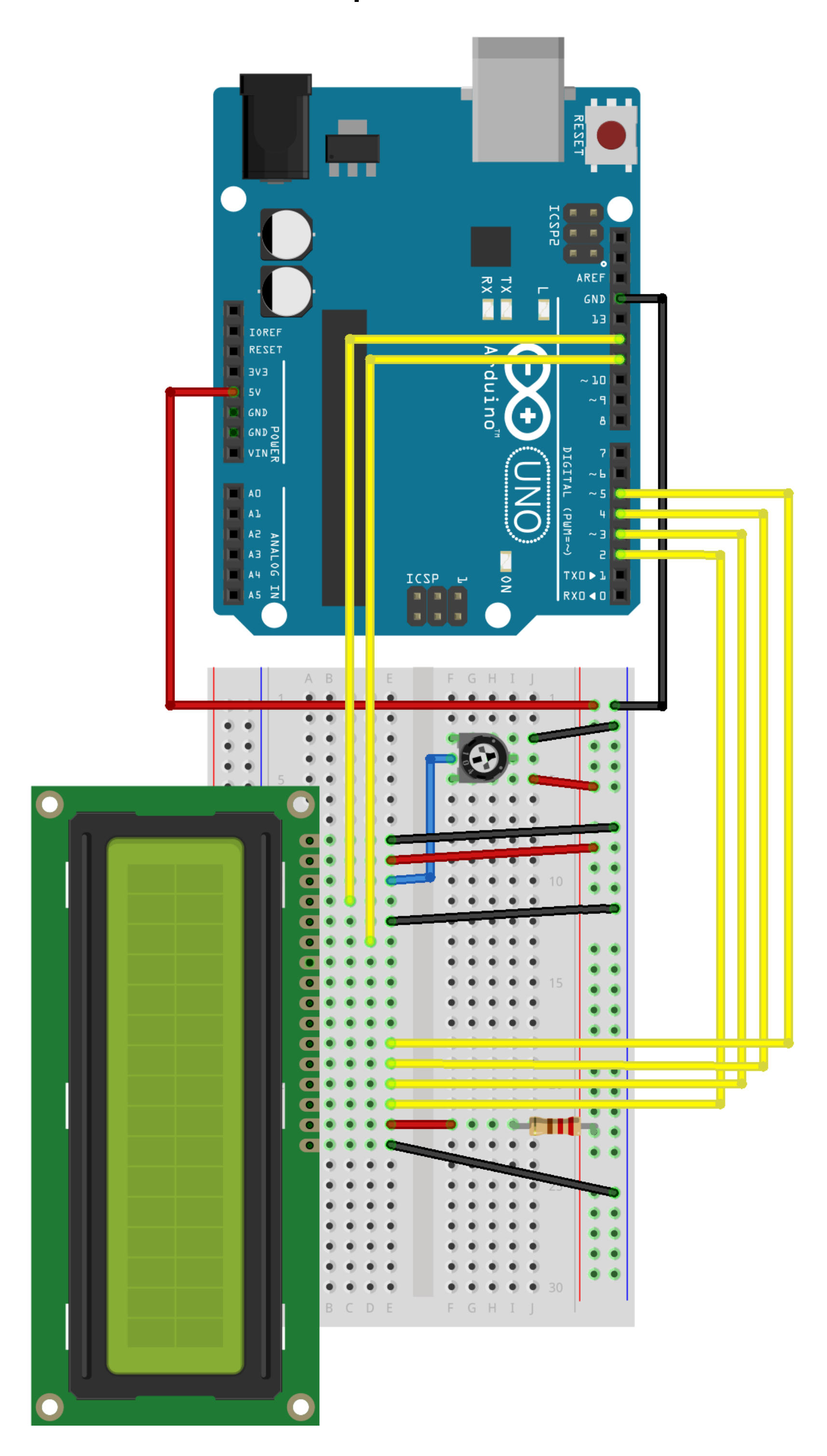
هذا المثال رسم تخطيطي "Hello World!" إلى LCD ويظهر الوقت بالثواني منذ إعادة تعيين Arduino.




تحتوي شاشات LCD على واجهة متوازية ، مما يعني أن وحدة التحكم الدقيقة تضطر إلى التعامل مع العديد من مسامير الواجهة في وقت واحد للتحكم في العرض. تتكون الواجهة من الدبابيس التالية:

| | |
|---|---|
| register select (RS) pin | display constrast pin (Vo) |
| Read/Write (R/W) pin | power supply pins (+5V and Gnd) |
| Enable pin | LED Backlight (Bklt+ and BKlt-) |
| 8 data pins (D0 -D7) | |

المخطط :


RESET
ICSP2
AREF
GND
13
~10
~9
8
7
~6
~5
4
~3
2
TX0 ▶ 1
RX0 ◀ 0
DIGITAL (PWM=~)
TX
RX
L
Arduino™
UNO
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
A B E F G H I J
B C D E F G H I J

الكود الرسمي من شركة اردوينو :

```
/*
  LiquidCrystal Library - Hello World

 Demonstrates the use a 16x2 LCD display.  The LiquidCrystal
 library works with all LCD displays that are compatible with the
 Hitachi HD44780 driver. There are many of them out there, and you
 can usually tell them by the 16-pin interface.

 This sketch prints "Hello World!" to the LCD
 and shows the time.

  The circuit:
 * LCD RS pin to digital pin 12
 * LCD Enable pin to digital pin 11
 * LCD D4 pin to digital pin 5
 * LCD D5 pin to digital pin 4
 * LCD D6 pin to digital pin 3
 * LCD D7 pin to digital pin 2
 * LCD R/W pin to ground
 * LCD VSS pin to ground
 * LCD VCC pin to 5V
 * 10K resistor:
 * ends to +5V and ground
 * wiper to LCD VO pin (pin 3)

 Library originally added 18 Apr 2008
 by David A. Mellis
 library modified 5 Jul 2009
 by Limor Fried (http://www.ladyada.net)
 example added 9 Jul 2009
 by Tom Igoe
 modified 22 Nov 2010
 by Tom Igoe
 modified 7 Nov 2016
 by Arturo Guadalupi
```

```
 This example code is in the public domain.

 http://www.arduino.cc/en/Tutorial/LiquidCrystalHelloWorld

*/

// include the library code:
#include <LiquidCrystal.h>

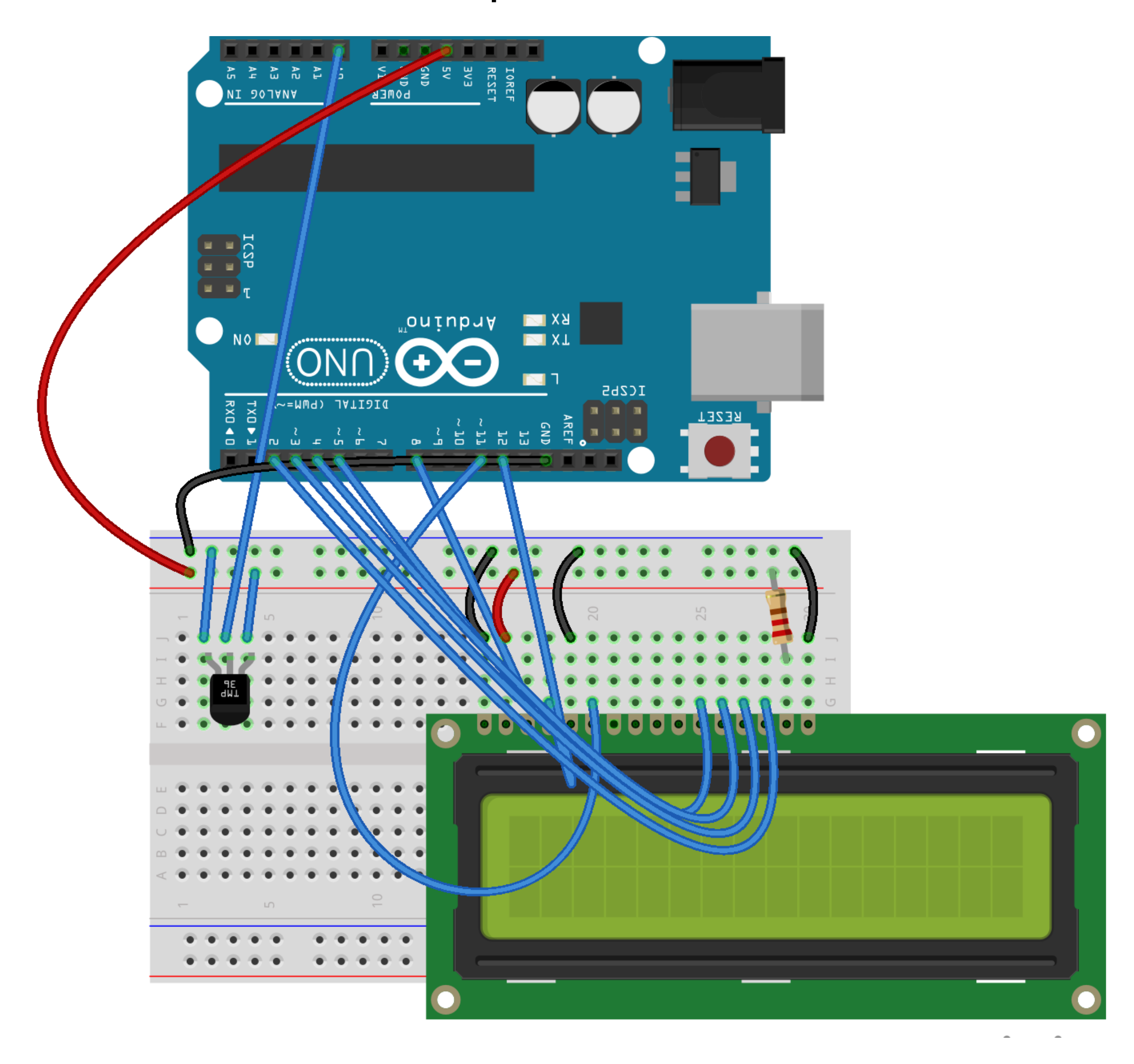
// initialize the library by associating any needed LCD interface
pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with
0):
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis() / 1000);
}
```

- عرض معلومات حساس درجة الحرارة على الشاشة

المخطط:



الكود:

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

const int tmp = A0;
const int p = 8;

void setup() {
```

```
  lcd.begin(16, 2);
  Serial.begin(9600);
  pinMode(p, OUTPUT);
  // put your setup code here, to run once:

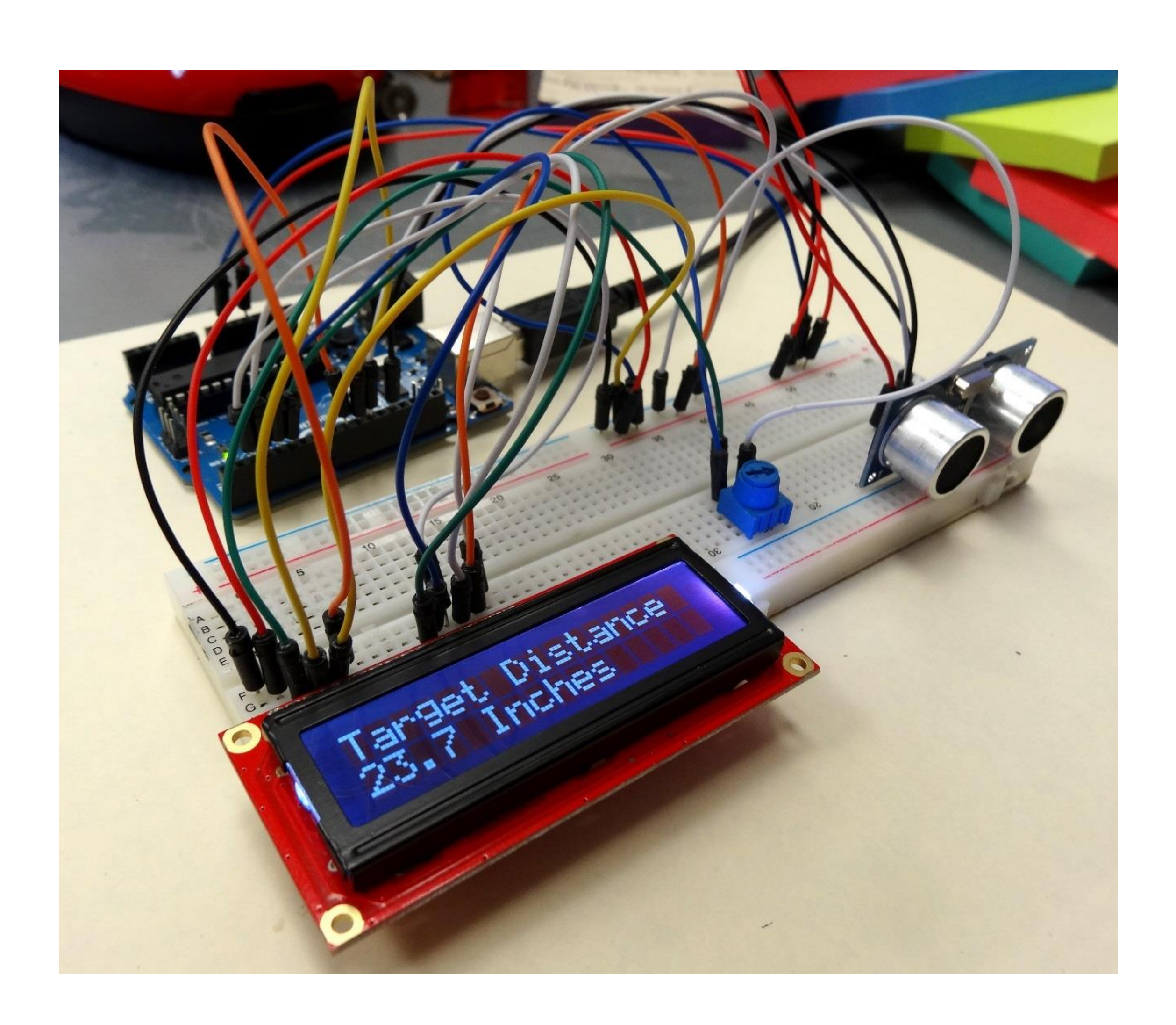
}

void loop() {
  digitalWrite(p,LOW);
  int Temp = analogRead(tmp);
  float volts = (Temp / 965.0) * 5;
  float c = (volts - .5) * 100;
  float f = (c * 9 / 5 + 32);
  Serial.println(f);
  lcd.setCursor(5, 0);
  lcd.print(f);
  delay(3000);
  // put your main code here, to run repeatedly:

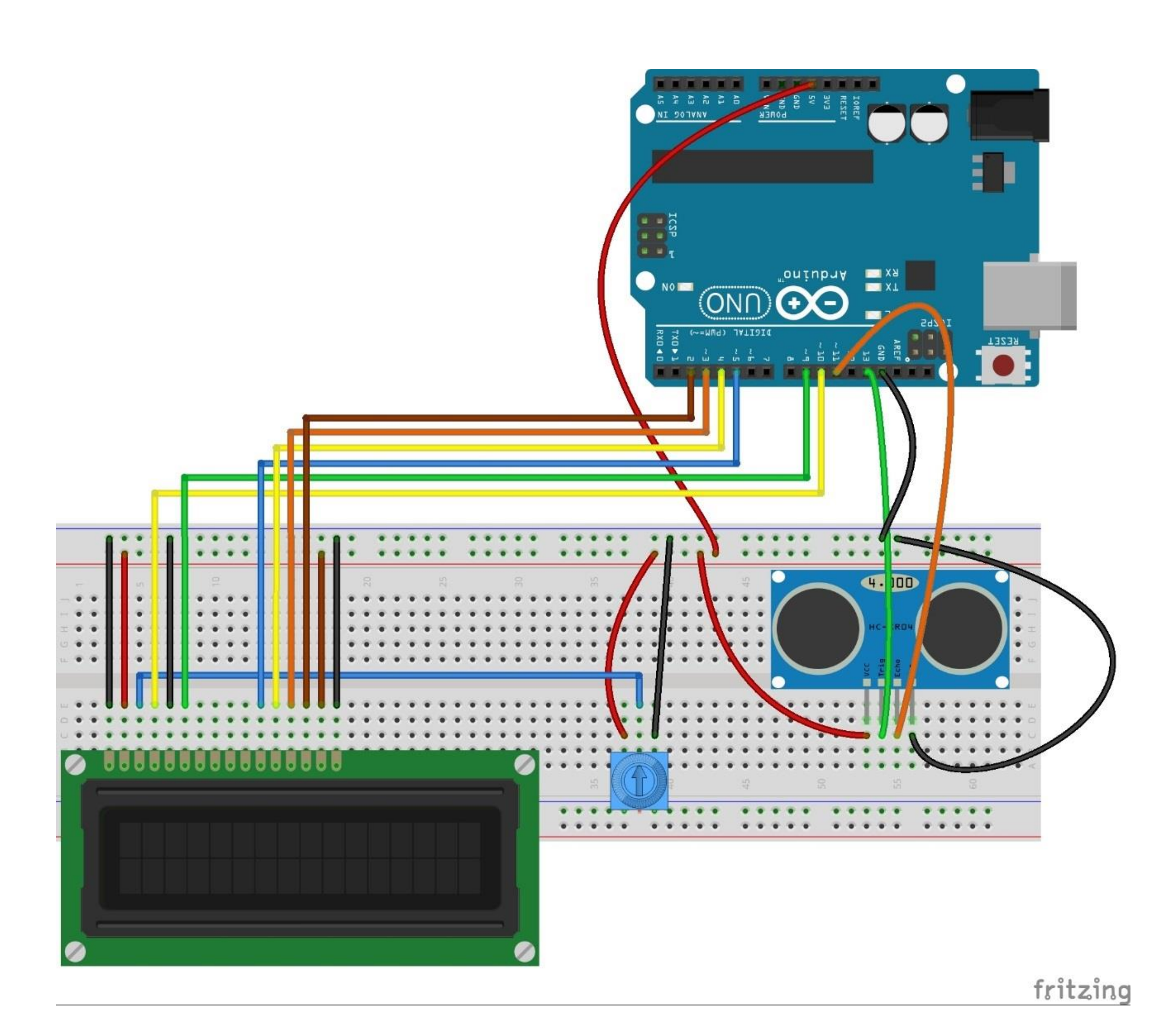
}
```

- اظهار المسافة على الشاشة

قمنا سابقا باستخدام حساس الامواج فوق الصوتية مع مصابيح LED في هذا المثال سوف نجعل الحساس يقوم بعملية القياس واظهار المسافة .



المخطط :




الكود :

```
#include <LiquidCrystal.h> //Load Liquid Crystal Library
LiquidCrystal LCD(10, 9, 5, 4, 3, 2);  //Create Liquid Crystal
Object called LCD

int trigPin=13; //Sensor Trip pin connected to Arduino pin 13
int echoPin=11;  //Sensor Echo pin connected to Arduino pin 11
int myCounter=0;  //declare your variable myCounter and set to 0
int servoControlPin=6; //Servo control line is connected to pin 6
```

```
float pingTime;  //time for ping to travel from sensor to target
and return
float targetDistance; //Distance to Target in inches
float speedOfSound=776.5; //Speed of sound in miles per hour when
temp is 77 degrees.

void setup() {

Serial.begin(9600);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);

LCD.begin(16,2); //Tell Arduino to start your 16 column 2 row LCD
LCD.setCursor(0,0);  //Set LCD cursor to upper left corner, column
0, row 0
LCD.print("Target Distance:");  //Print Message on First Row
}

void loop() {

  digitalWrite(trigPin, LOW); //Set trigger pin low
  delayMicroseconds(2000); //Let signal settle
  digitalWrite(trigPin, HIGH); //Set trigPin high
  delayMicroseconds(15); //Delay in high state
  digitalWrite(trigPin, LOW); //ping has now been sent
  delayMicroseconds(10); //Delay in high state

  pingTime = pulseIn(echoPin, HIGH);  //pingTime is presented in
microceconds
  pingTime=pingTime/1000000; //convert pingTime to seconds by
dividing by 1000000 (microseconds in a second)
  pingTime=pingTime/3600; //convert pingtime to hourse by dividing
by 3600 (seconds in an hour)
  targetDistance= speedOfSound * pingTime;  //This will be in
miles, since speed of sound was miles per hour
  targetDistance=targetDistance/2; //Remember ping travels to
target and back from target, so you must divide by 2 for actual
target distance.
```

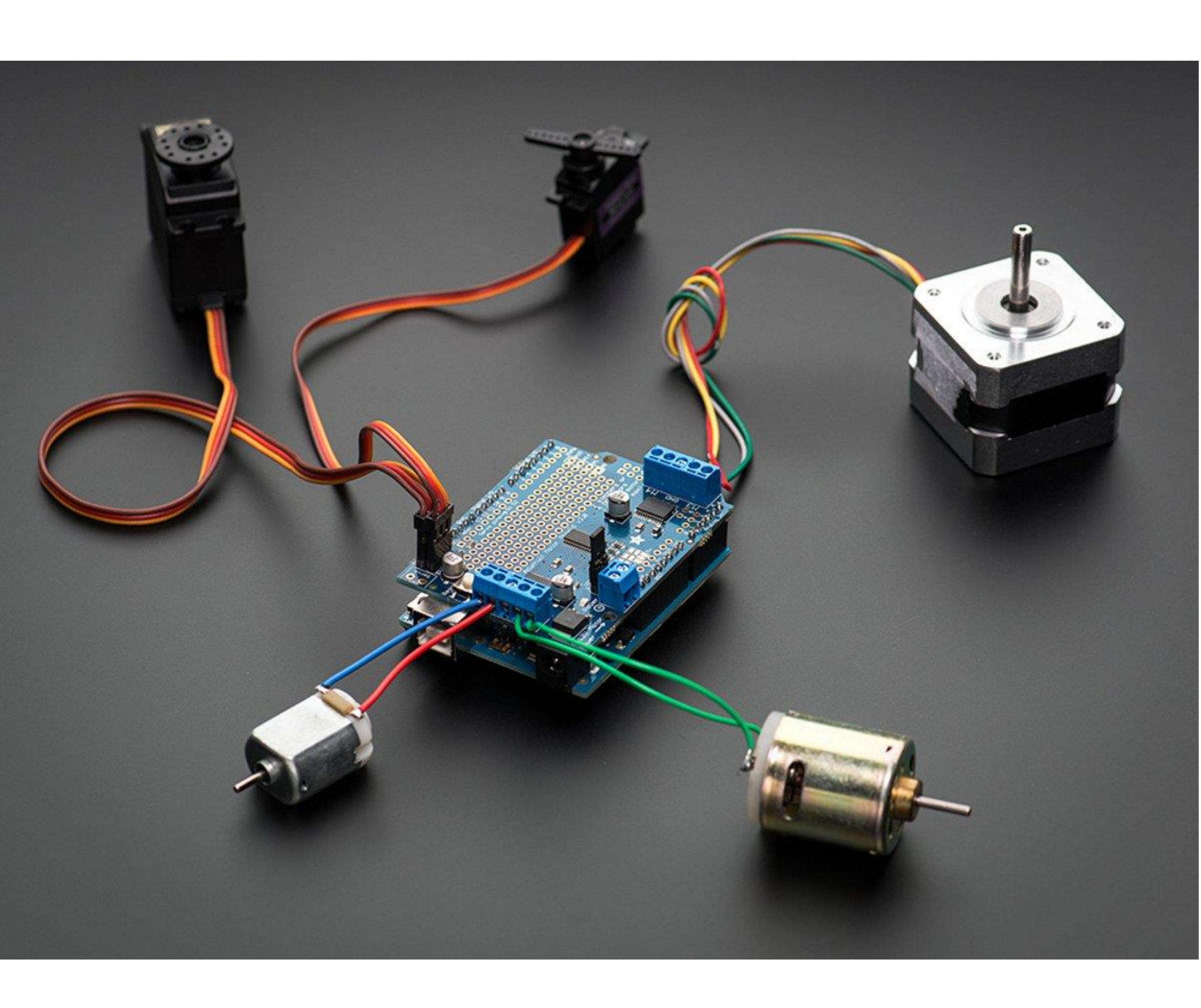
```
  targetDistance= targetDistance*63360;    //Convert miles to
inches by multipling by 63360 (inches per mile)

  LCD.setCursor(0,1);  //Set cursor to first column of second row
  LCD.print("                "); //Print blanks to clear the row
  LCD.setCursor(0,1);   //Set Cursor again to first column of
second row
  LCD.print(targetDistance); //Print measured distance
  LCD.print(" inches");  //Print your units.
  delay(250); //pause to let things settle



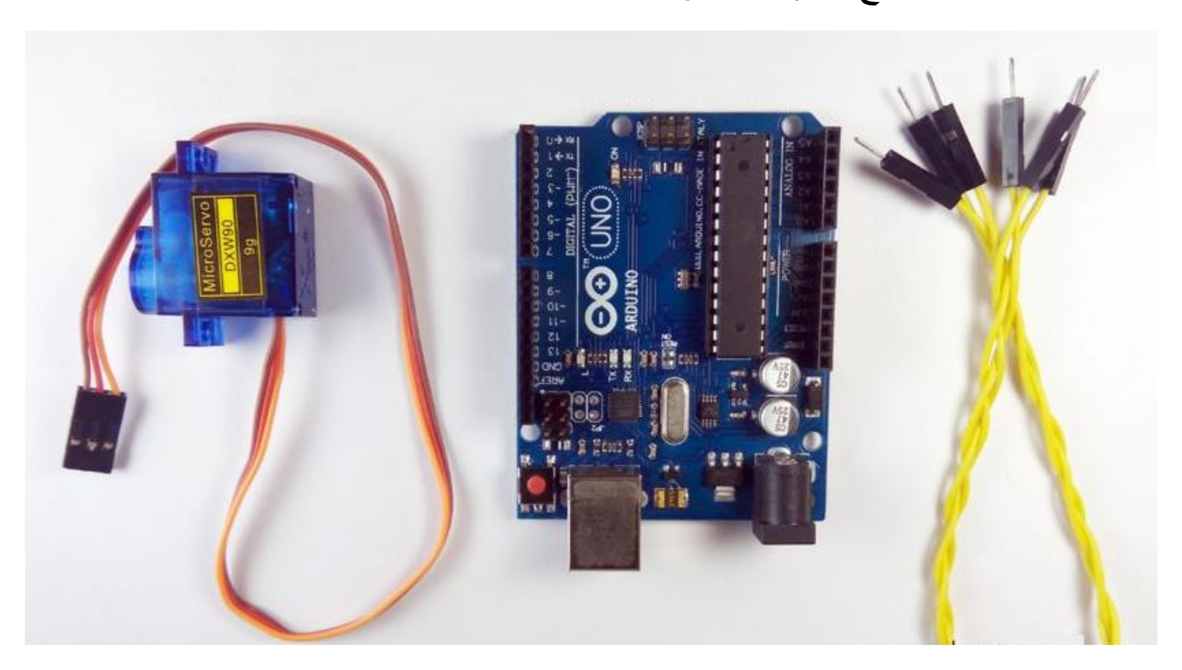
  }
```

## المحركات Motors



لقد درسنا انواع المحركات الكهربائية في الجزء الثاني من السلسلة حيث تعرفنا على انواع المحركات المختلفة في هذا الدرس سوف نبدأ في استخدام المحركات المختلفة باستخدام المتحكم الالكتروني اردوينو , هنا تصبح المشاريع مليئة بالحيوية وممتلئة بالافكار العتادية التفاعلية .

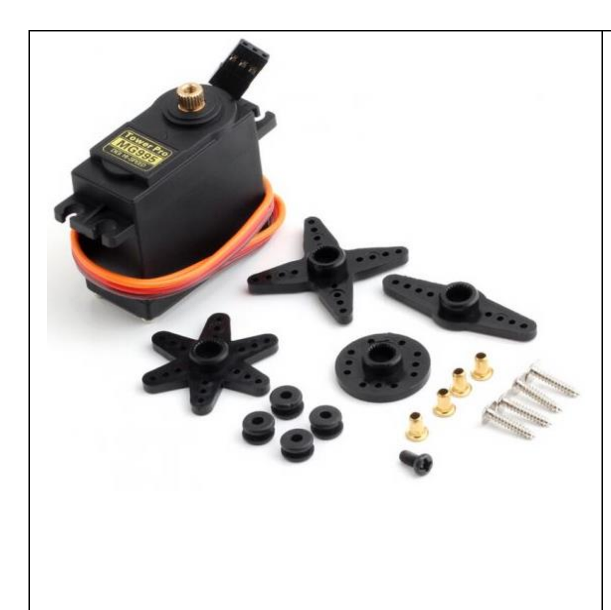
- اولا التعامل مع محرك السيرفو




لقد درسنا سابقا ان محرك السرفو يعمل الدرجات فهو لا يتحرك بشكل دائم مثل محرك ال التيار المستمر DC او غيرها من المحركات , يمكن برمجة السيرفو على اساس الحركة مع او عكس عقارب الساعة بقيمة معينة فقط فبعض محركات السيرفو يتحرك180 درجة والبعض 270 و 360, والجدول التالي يبين الانواع المختلفة لمحرك السيرفو:

| | |
|---|---|
|  | MINI SERVO MOTOR<br>SG90 9G<br>Features:<br>Good quality & competitive price<br>Dimension: 23x12.2x29mm<br>Torque: 0.5kg/cm<br>Stall torque: 1.5kg/cm at 5V<br>Operating speed: 0.3sec/60degree at 4.8V<br>Operating voltage: 4.2-6V<br>Temperature range: 0-55 °C<br>Dead band width: 10us<br>Gear medium: Nylon<br>Brown cable: Negative<br>Red cable: Positive<br>Orange cable: Signal wire |

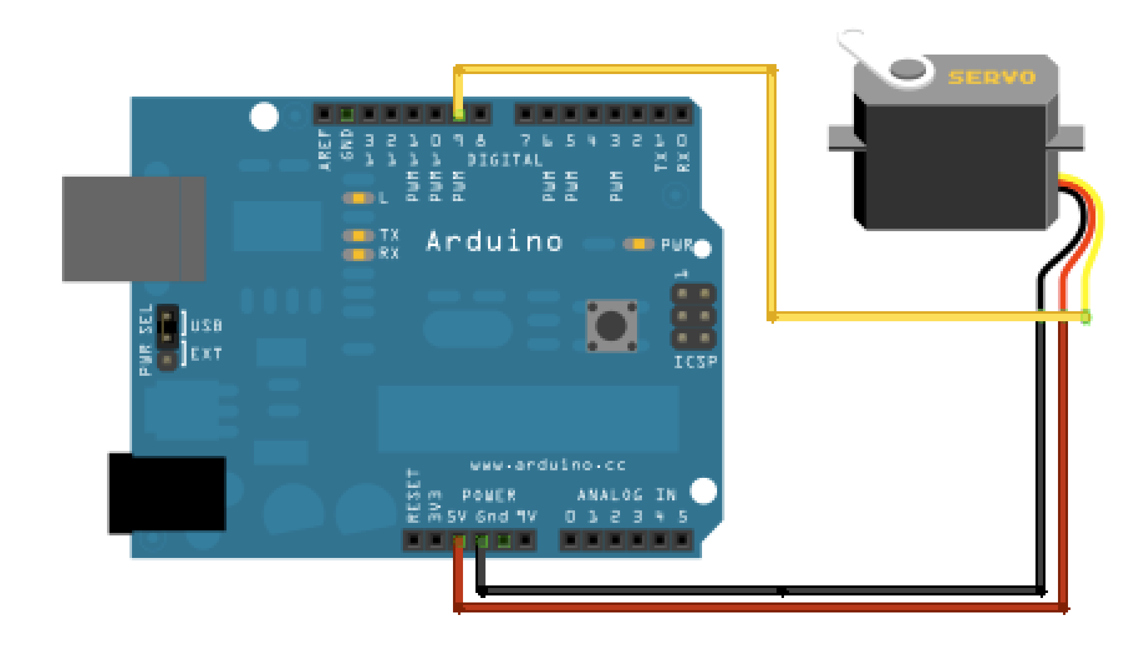
| | |
|---|---|
| <br><br> | SERVO MOTOR MG995 METAL GEAR 180 DEGREE<br><br>Features:<br><br>Compliant with most standard receiver connector: FutabaHitecSanwaGWS etc...<br>Great for truckBoatRacing CarHelicopter and Airplane.<br>Power Supply: Through External Adapter.<br>Stable and Shock Proof. - Coreless Motor - Metal Gear - Double Ball Bearing - Connector Wire Length 300mm<br>Dimension : 40mm x 19mm x 43mm<br>Weight : 55g<br>Operating Speed : 0.17sec / 60 degrees (4.8V no load)<br>Operating Speed : 0.13sec / 60 degrees (6.0V no load)<br>Stall Torque : 13 kg-cm (180.5 oz-in) at 4.8V<br>Stall Torque : 15 kg-cm (208.3 oz-in) at 6V<br>Operation Voltage : 4.8 - 7.2Volts<br>Gear Type: All Metal Gears<br>Connector Wire: Heavy Duty11.81" (300mm)<br>Angle: 180 degree |
| <br><br> | POWER HD HIGH-TORQUE SERVO 17KG<br><br>Description:<br><br>Digital?: N<br>Speed @ 6V: 0.14 sec/60°<br>Stall torque @ 6V: 17 kg·cm<br>Speed @ 4.8V: 0.16 sec/60°<br>Stall torque @ 4.8V: 15.5 kg·cm<br>Lead length: 11 in<br>Hardware included?: Y<br>Dimensions<br>Size: 40.7 x 20.5 x 39.5 mm<br>Weight: 60g |

| | |
|---|---|
|  | DEGREE SERVO 360<br>MG995<br><br>Description:<br><br>This TowerPro 360 Degree Servo with 13kg/cm stall torque is a good choice for your RC airplanes. It comes with servo arms and screws for easy installation.<br><br>Specification:<br><br>Brand: TowerPro.<br>Model: MG995.<br>Type: Metal 360 Degree Servo<br>Dimension: 5.4 cm x 4.4 cm x 2.0 cm.<br>Stall Torque (4.8V):13kg/cm<br>Stall Torque(6.0V):15kg/cm<br>Speed (4.8v):0.17sec/360degrees<br>Speed (6V):0.13sec/360degrees<br>Operating voltage: 4.2-6V.<br>Temperature range: 055.<br>Weight: 48g |

بعد ان تعرفنا على انواع السيرفو المختلفة سوف نقوم الان في برمجة محرك السيرفو ليعمل مسح (Sweep) او ان يتحرك ذهابا وإيابا عبر 180 درجة.

تشتمل محركات المؤازرة (السيرفو) على ثلاثة أسلاك: الطاقة والأرضي والإشارة. عادة ما يكون سلك الطاقة باللون الأحمر ، ويجب توصيله بالمسمار 5V الموجود على لوح Arduino. السلك الأرضي هو عادة أسود أو بني ويجب توصيله بدبوس أرضي على اللوح. دبوس الإشارة هو عادة أصفر أو برتقالي أو أبيض ويجب توصيله بالرقم 9 الرقمي على اللوحة في هذا المثال .

المخطط :




الكود :

```
/* Sweep
 by BARRAGAN <http://barraganstudio.com>
 This example code is in the public domain.

 modified 8 Nov 2013
 by Scott Fitzgerald
 http://www.arduino.cc/en/Tutorial/Sweep
*/

#include <Servo.h>

Servo myservo;  // create servo object to control a servo
// twelve servo objects can be created on most boards

int pos = 0;    // variable to store the servo position

void setup() {
```

```
  myservo.attach(9);  // attaches the servo on pin 9 to the servo
object
}

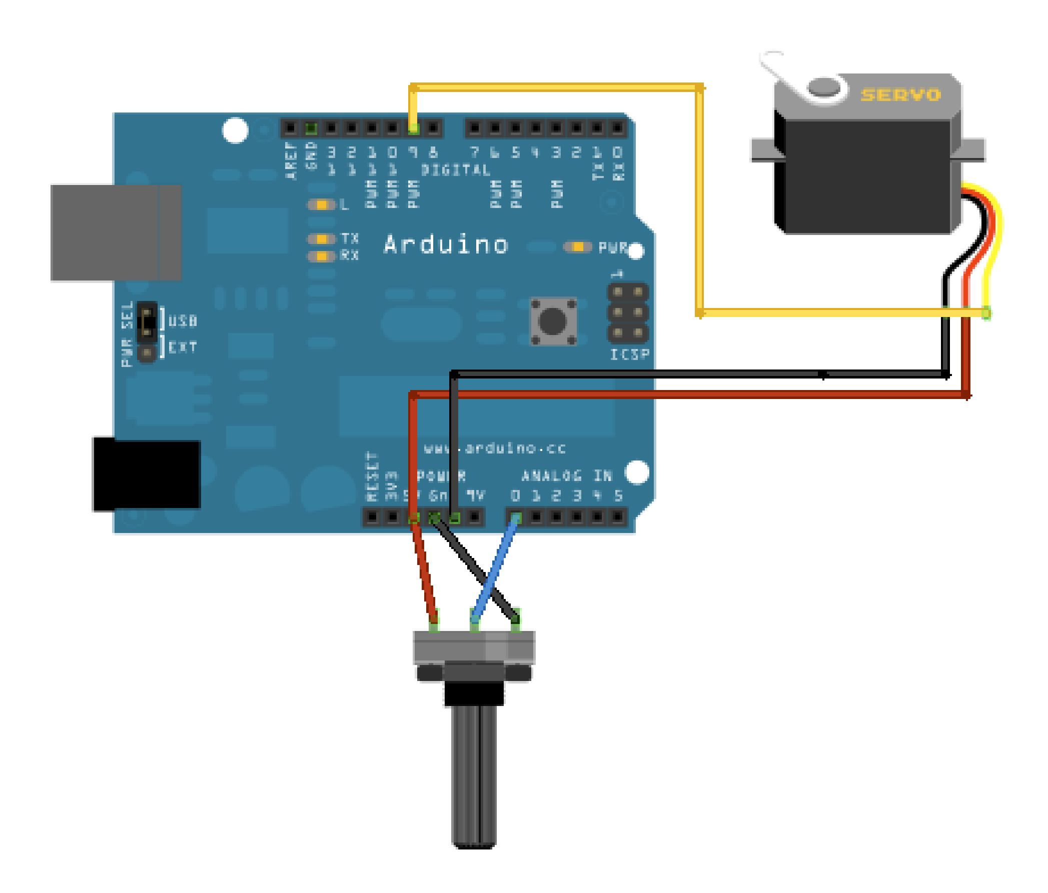
void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to
180 degrees
    // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to
position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to
reach the position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to
0 degrees
    myservo.write(pos);              // tell servo to go to
position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to
reach the position
  }
}
```

ستلاحظ ان محرك السيرفو بدء التحرك يمينا وشمالا.

- التحكم بحركة محرك السيرفو باستخدام مجزء الجهد/مقاومة متغيرة (Knob)

يكون سلك الطاقة باللون الأحمر ، ويجب توصيله بالمسمار V5 الموجود على لوح Arduino. السلك الأرضي هو عادة أسود أو بني ويجب توصيله بدبوس أرضي على اللوح. دبوس الإشارة هو عادة أصفر أو برتقالي أو أبيض ويجب توصيله بالرقم 9 الرقمي على اللوحة في هذا المثال , يجب أن يكون مجزء(potentiometer) الجهد سلكيًا بحيث يتم توصيل مساميره الخارجية بالطاقة (+ 5 فولت) والأرض GND ، ويتم توصيل دبوسها الأوسط بالمدخل التناظري a0 على اللوحة.

المخطط :



الكود :

```
#include <Servo.h>

Servo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
int val;    // variable to read the value from the analog pin

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo
object
}

void loop() {
  val = analogRead(potpin);            // reads the value of the
potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 180);     // scale it to use it with
the servo (value between 0 and 180)
  myservo.write(val);                  // sets the servo position
according to the scaled value
  delay(15);                           // waits for the servo to
get there
}
```

قم بتحريك المقاومة المتغيرة وستلاحظ تغير حركة المحرك يدويا

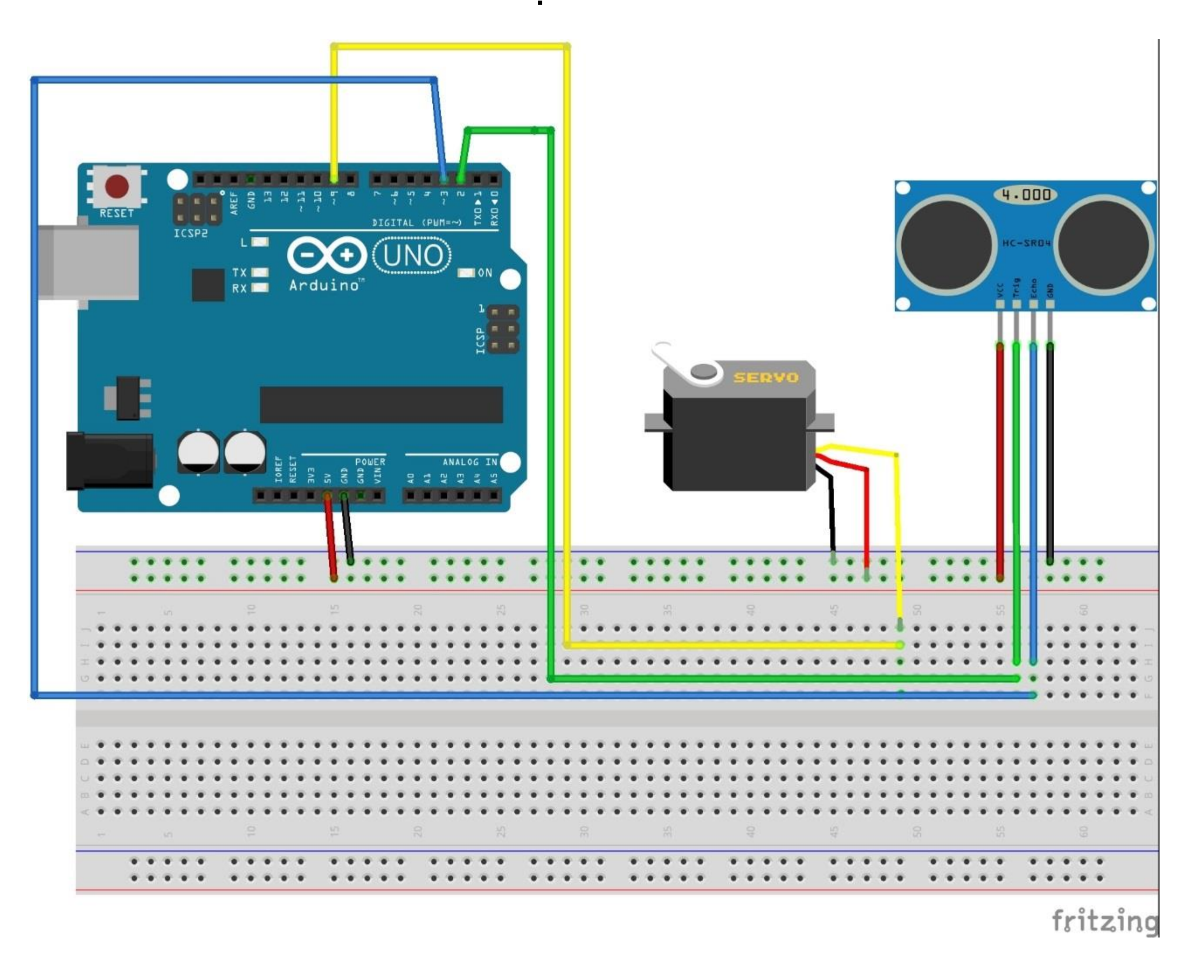
حسنا نريد اضافة مزيدا من التفاعل بين مكونات الاردوينو سوف ندرس برنامج جديد من برامج هذه السلسلة وهو برنامج processing يمكننا هذا البرنامج من عمل مشاريع كبيرة ووتفاعلية جدا باستخدام للغة الجافا JAVA

يمكن تحميل البرنامج من خلال الرابط التالي :

/https://processing.org/download

## رادار

المخطط :



## كود الاردوينو

```
#include<Servo.h>
int trigPin=2;
int echoPin=3;

long duration;
int distance;
Servo servo;

void setup()
{
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  servo.attach(9);
}

void loop()
{
  for(int i=15;i<=165;i++)
  {
    servo.write(i);
    delay(100);
    distance=calculateDistance();

    Serial.print(i);
    Serial.print(",");
    Serial.print(distance);
    Serial.print(".");
  }
  for(int i=165;i>15;i--)
  {
    servo.write(i);
    delay(100);
    distance=calculateDistance();

    Serial.print(i);
```

```
    Serial.print(",");
    Serial.print(distance);
    Serial.print(".");
  }
}

int calculateDistance()
{
  digitalWrite(trigPin,LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin,LOW);

  duration=pulseIn(echoPin,HIGH);
  distance=duration*0.034/2;
  return distance;
}
```

## كود المعالج بلغة الجافا

```
import processing.serial.*;
import java.awt.event.KeyEvent;
import java.io.IOException;



Serial myPort;
String angle="";
String distance="";
String data="";
String noObject;
float pixelsDistance;
int iAngle,iDistance;
int index1=0,index2=0;
PFont orcFont;

void setup()
```

```
{
  size(1496,900);
  smooth();
  myPort=new Serial(this,"COM3",9600);
  myPort.bufferUntil('.');
  orcFont=loadFont("OCRAExtended-30.vlw");
}

void draw()
{
  fill(98,245,31);
  textFont(orcFont);
  noStroke();
  fill(0,4);
  rect(0,0,width,1010);

  fill(98,245,31);

  drawRadar();
  drawLine();
  drawObject();
  drawText();
}

void serialEvent(Serial myPort)
{
  data=myPort.readStringUntil('.');
  data=data.substring(0,data.length()-1);

  index1=data.indexOf(",");
  angle=data.substring(0,index1);
  distance=data.substring(index1+1,data.length());

  iAngle=int(angle);
  iDistance=int(distance);
}

void drawRadar(){
  pushMatrix();
```

```
  translate(760,800);
  noFill();
  strokeWeight(2);
  stroke(98,245,31);
  arc(0,0,1400,1400,PI,TWO_PI);
  arc(0,0,1100,1100,PI,TWO_PI);
  arc(0,0,800,800,PI,TWO_PI);
  arc(0,0,500,500,PI,TWO_PI);
  line(-760,0,760,0);
  line(0,0,-760*cos(radians(30)),-760*sin(radians(30)));
  line(0,0,-760*cos(radians(60)),-760*sin(radians(60)));
  line(0,0,-760*cos(radians(90)),-760*sin(radians(90)));
  line(0,0,-760*cos(radians(120)),-760*sin(radians(120)));
  line(0,0,-760*cos(radians(150)),-760*sin(radians(150)));
  line(-760*cos(radians(30)),0,760,0);
  popMatrix();
}


void drawObject() {
  pushMatrix();
  translate(760,800); // moves the starting coordinats to new
location
  strokeWeight(9);
  stroke(255,10,10); // red color
  pixelsDistance = iDistance*22.5; // covers the distance from the
sensor from cm to pixels
  // limiting the range to 40 cms
  if(iDistance<40){
    // draws the object according to the angle and the distance
  line(pixelsDistance*cos(radians(iAngle)),-
pixelsDistance*sin(radians(iAngle)),750*cos(radians(iAngle)),-
750*sin(radians(iAngle)));
  }
  popMatrix();
}

void drawLine()
{
```

```
  pushMatrix();
  strokeWeight(9);
  stroke(30,250,60);
  translate(760,800);
  line(0,0,750*cos(radians(iAngle)),-750*sin(radians(iAngle)));
  popMatrix();

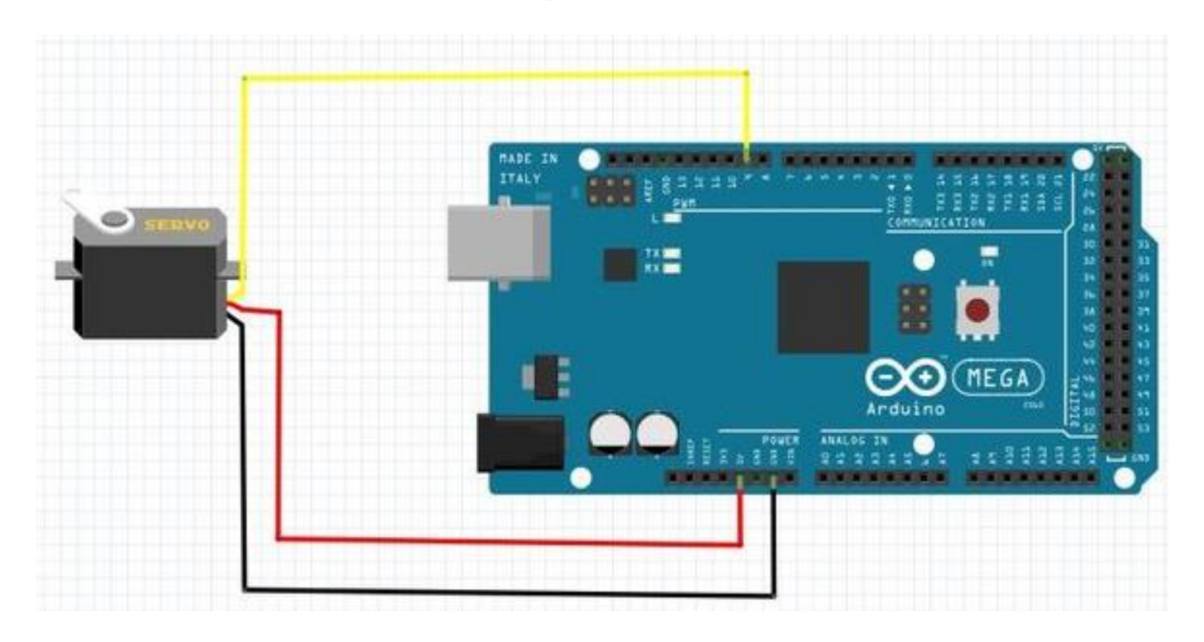
}

void drawText()
{
  pushMatrix();
  if(iDistance>40)
  {
    noObject="Out ofRange";
  }
  else{
    noObject="In Range";
  }
  fill(0,0,0);
  noStroke();
  rect(0, 1010, width, 1080);
  fill(98,245,31);
  textSize(25);
  text("10cm",1180,990);
  text("20cm",1380,990);
  text("30cm",1580,990);
  text("40cm",1780,990);
  textSize(30);
  text("Object: " + noObject,15,100);
  text("Angle: " + iAngle +" Â°",400, 50);
  text("Distance: ", 13,50);
  if(iDistance<40) {
  text("        " + iDistance +" cm",30,50);
  }
  textSize(15);
  fill(98,245,60);
  translate(761+760*cos(radians(30)),782-760*sin(radians(30)));
  rotate(-radians(-60));
```

```
  text("30Â°",0,0);
  resetMatrix();
  translate(754+760*cos(radians(60)),784-760*sin(radians(60)));
  rotate(-radians(-30));
  text("60Â°",0,0);
  resetMatrix();
  translate(745+760*cos(radians(90)),790-760*sin(radians(90)));
  rotate(radians(0));
  text("90Â°",0,0);
  resetMatrix();
  translate(735+760*cos(radians(120)),803-760*sin(radians(120)));
  rotate(radians(-30));
  text("120Â°",0,0);
  resetMatrix();
  translate(740+760*cos(radians(150)),818-760*sin(radians(150)));
  rotate(radians(-60));
  text("150Â°",0,0);
  popMatrix();
}
```

التحكم في سرعة محرك السيرفو بواسطة نافذة ال serial monitor

المخطط :




الكود :

```
//control servo motor with serial monitering window - by ujash
patel
  //for more go to www.uu-machinetool.blogspot.com
#include <Servo.h>

Servo myservo;  // create servo object to control a servo
                // a maximum of eight servo objects can be created

int pos = 0;
int servodata;

void setup()  {
 Serial.begin(9600);
Serial.println("Redy");
Serial.println("select speed 1 for 15ms delay ");
```

```
Serial.println("select speed 2 for 25ms delay");
Serial.println("select speed 3 for 5ms delay");
  myservo.attach(9);

}

void loop()
{
  if (Serial.available() > 0)
  {
   servodata = Serial.read();

    if(servodata == '1') // Single Quote! This is a character.
    {
      Serial.println("speed 1 is selected");
      Serial.println("DELAY 15MS");
     for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to
180 degrees
  {                                  // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to
position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to
reach the position
  }
  for(pos = 180; pos>=1; pos-=1)     // goes from 180 degrees to 0
degrees
  {
    myservo.write(pos);              // tell servo to go to
position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to
reach the position
  }
    }

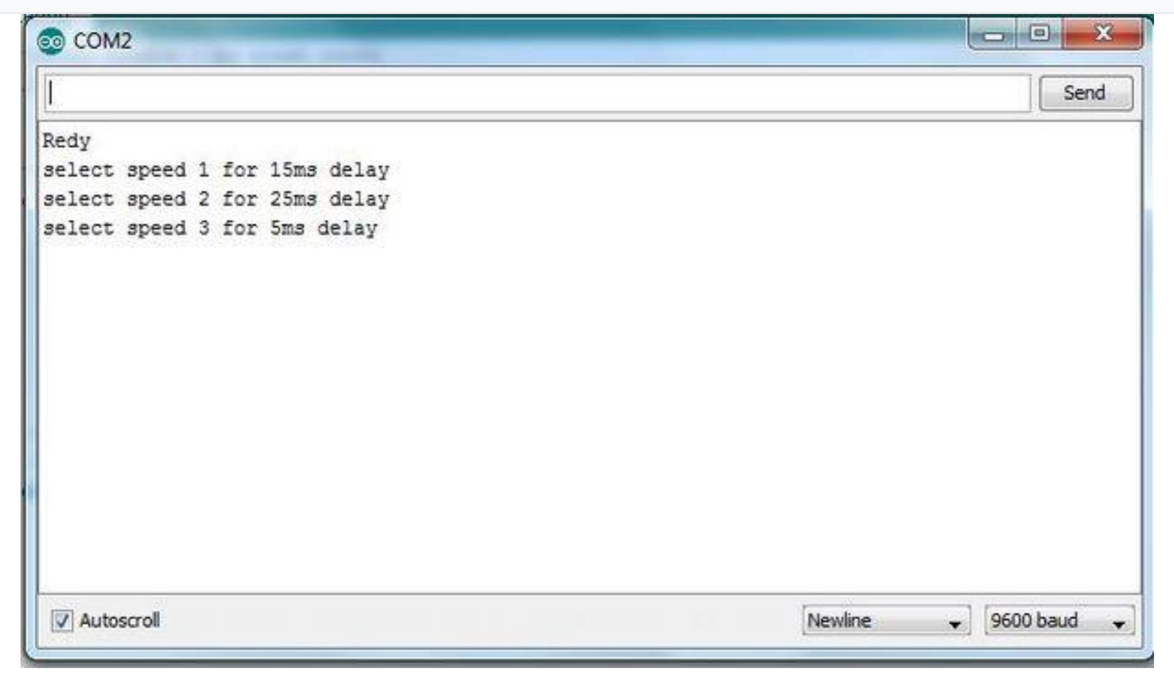
    if(servodata== '2')
    {
      Serial.println("speed 2 is selected");
      Serial.println("DELAY 25MS ");
```

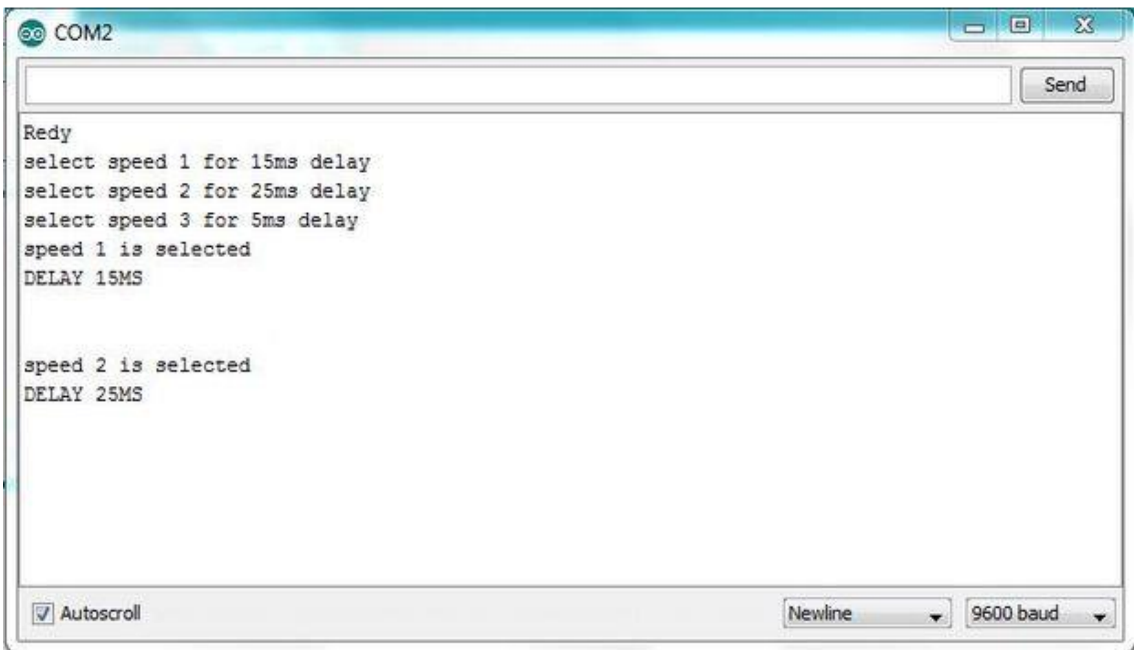
```
     for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to 
180 degrees 
  {                                  // in steps of 1 degree 
    myservo.write(pos);              // tell servo to go to 
position in variable 'pos' 
    delay(25);             // waits 15ms for the servo to reach the 
position 
  } 
  for(pos = 180; pos>=1; pos-=1)     // goes from 180 degrees to 0 
degrees 
  {                                
    myservo.write(pos);              // tell servo to go to 
position in variable 'pos' 
    delay(25);               // waits 15ms for the servo to reach 
the position 
  } 
    }
   if(servodata == '3')
    {
      Serial.println("speed 3 is selected");
      Serial.println("DELAY 5MS ");
     for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to 
180 degrees 
  {                                  // in steps of 1 degree 
    myservo.write(pos);              // tell servo to go to 
position in variable 'pos' 
    delay(5);         // waits 15ms for the servo to reach the 
position 
  } 
  for(pos = 180; pos>=1; pos-=1)     // goes from 180 degrees to 0 
degrees 
  {                                
    myservo.write(pos);              // tell servo to go to 
position in variable 'pos' 
    delay(5);              // waits 15ms for the servo to reach the 
position 
  } 
    }
```

```
    Serial.println();     // End the line

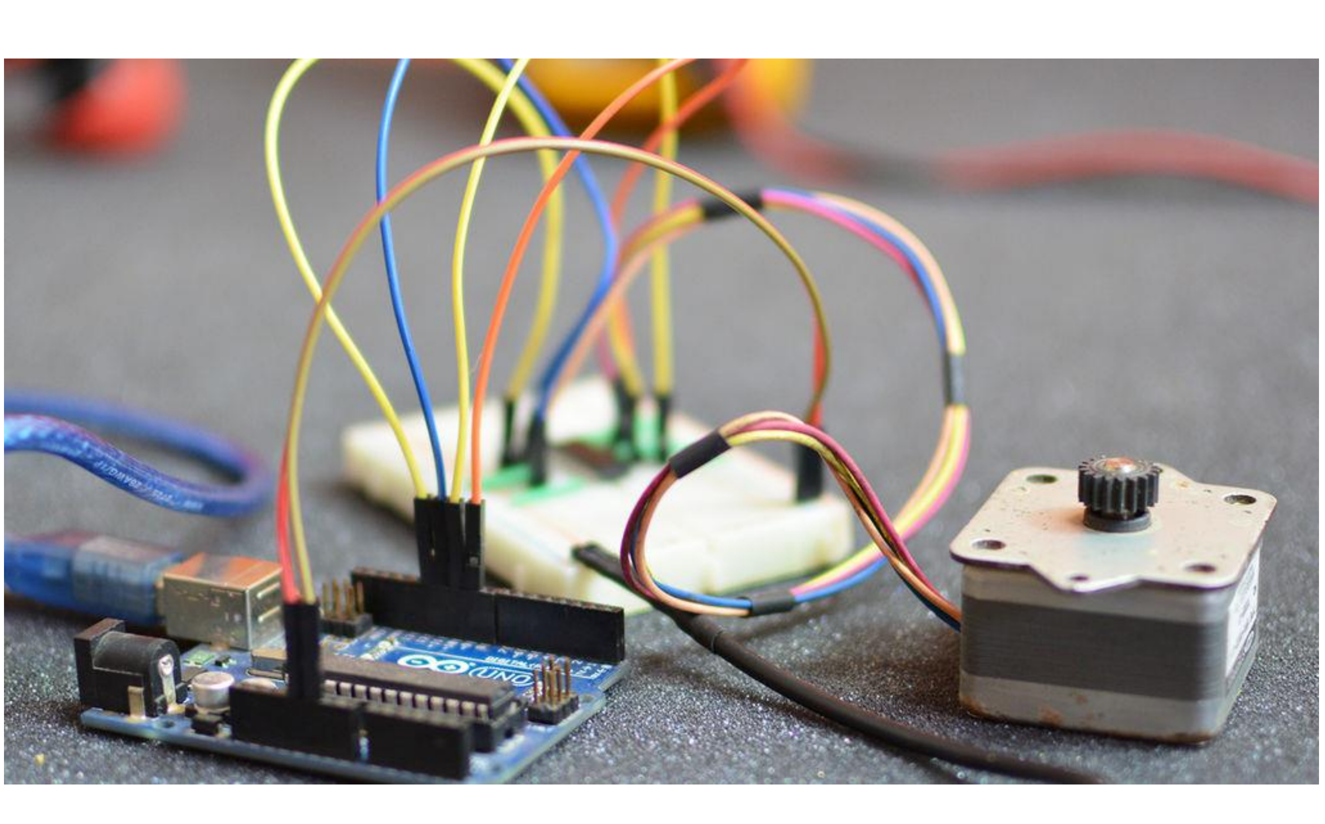


  }
}
```



- ثانيا التعامل مع المحرك الخطوي stepper motor



محرك Stepper motor ويدعى بالعربية المحرك الخطوي ( اي محرك خطوة بخطوة) وهو محرك كهربائي يستخدم في الآلات الصغيرة التي تحتاج لدقة في التحكم مثل الطابعة وقاطع الليزر والطابعات ثلاثية الابعاد وفي التطبيقات الروبوتية, من أهم مميزات هذا المحرك هو انه يمكنه التحكم في عدد وسرعة دوراته وزاوية التوقف بدقة.

لذلك يعتبر المحرك الخطوي ثورة في عالم الحركة والمحركات حيث انه يستطيع الحركة بدقة كبيرة ومهما اختلفت السرعة فهو يعمل بنفس القوة ونظرا لاستخداماته الكثيرة فهو يصلح في الاستخدام بدلا عن اي نوع من المحركات المختلفة كما يمكنك ايضا استخدامه بدلا من محرك الزاوية السيرفو , في هذا الدرس ستتعلم كيفية التحكم بمحرك stepper motor عبر الأردوينو ولوحة التحكم الخاصة به.

- المیزات :

1 - زاوية الدوران تتناسب مع نبضات الدخل.

2 - اكبر عزم دوران المحرك عند التوقف الكامل (إذا كانت الملفات منشطة اى بها تيار) .

3 - دقة فى تحديد المواقع خاصة بعد انتاج محركات جيدة تتمتع بدقة 3 - 5 ٪ من الخطوة وهذا الخطأ غير تراكمى من خطوة إلى أخرى.

4 - استجابة ممتازة للبدء / وللإيقاف / ولعكس الدوران.

5 - درجة فعالية (تسمى وثوقية ) مرتفعة جدا حيث لا توجد فرش كربونية بالمحرك ( هو نوع من انواع المحركات بدون الفرش الكربونية ). ولذلك فإن عمر المحرك ببساطة يعتمد على عمر محاور الدوران. bearing

6 - استجابة المحركات لنبضات الدخل الرقمية توفر امكانية التحكم باستخدام الحلقة المفتوحة open-loop مما يجعل المحرك أكثر بساطة وأقل تكلفة فى التحكم

7 - من الممكن تحقيق (الحصول على) سرعة دوران منخفضة جدا متزامنه مع الحمل المرتبط مباشرة بعامود الدوران.

8 - يمكن تحقيق مدى واسع من السرعات الدورانية حيث تتناسب السرعة مع تردد نبصات الدخل.

- العيوب :

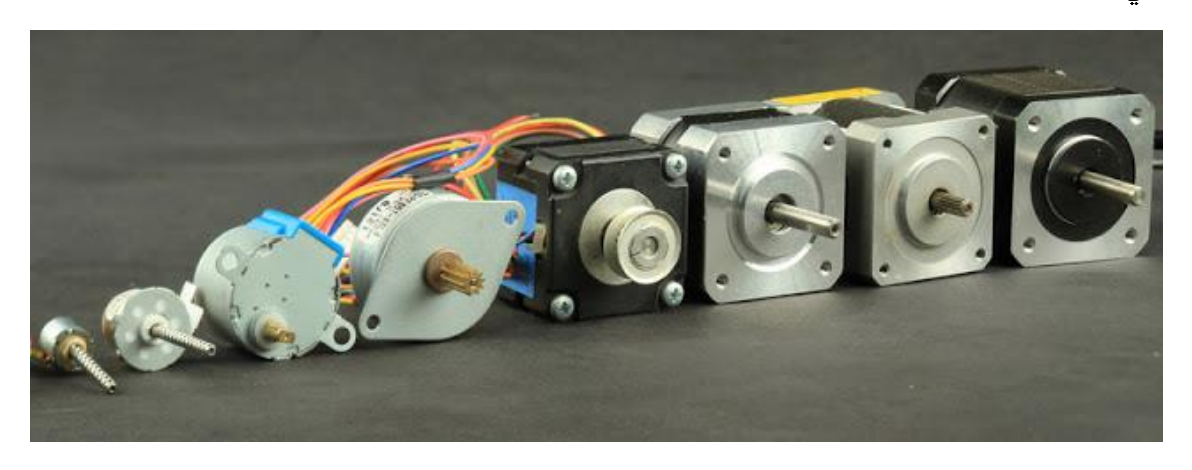
1 - يمكن حدوث اهتزازات او ما يعرف بالرنين او الأصداء Resonances اذا كان التحكم فيها غير صحيح او غير دقيق.

2 - ليس من السهل أن تعمل عند السرعات العالية جدا.

3 - تستهلك التيار عندما لا تتحرك (تستجر الطاقة في حالة اللا عمل )

4- باهظة الثمن نسبيا

في الصورة التالية بعض من اشكال المحركات الخطوية :



تعمل المحركات الخطوية عن طريق النبضات الكهربائية المتتابعة او عن طرق النبضات المنعكسة ونقوم بذلك باستخدام درايفر(متحكم) للتحكم بها

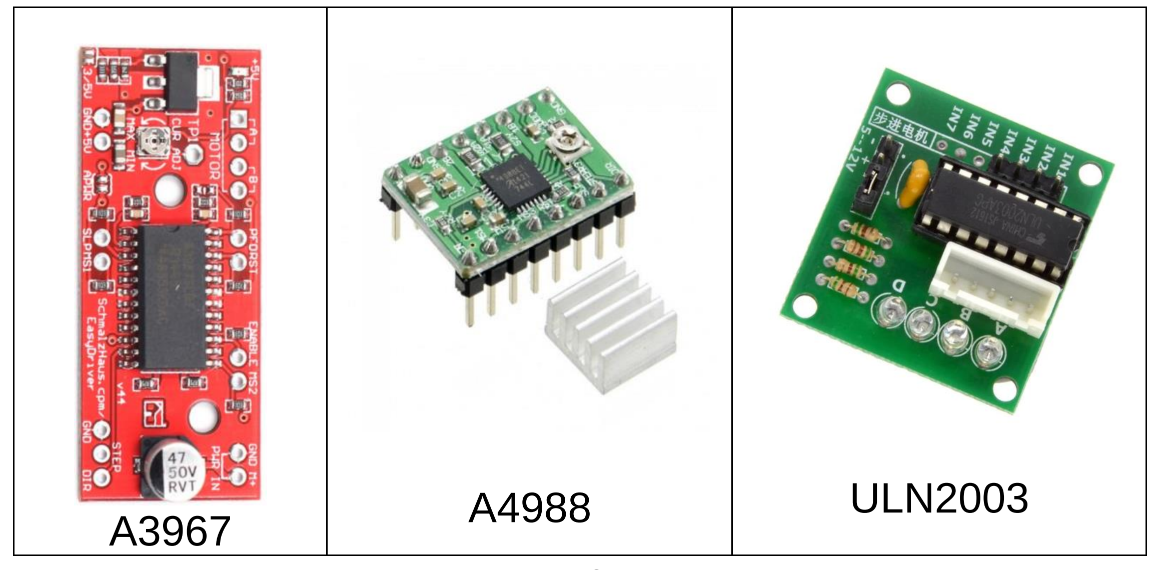
قيادة المحرك الخطوي:

هناك نوعين لدارات القيادة

1-دارة قيادة full/half step

2-دارة قيادة Micro Step

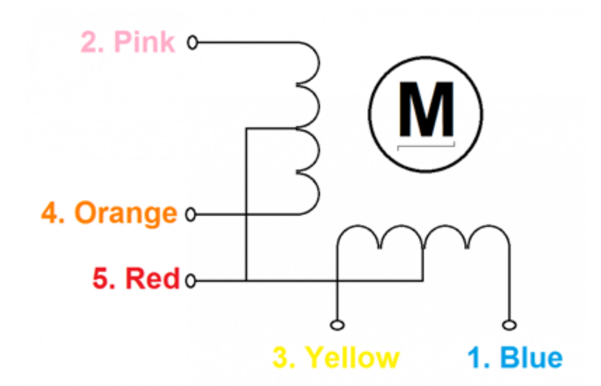
بعض من انواع الدرايفرات التي يمكننا استخدامها للتحكم في المحرك الخطوي:

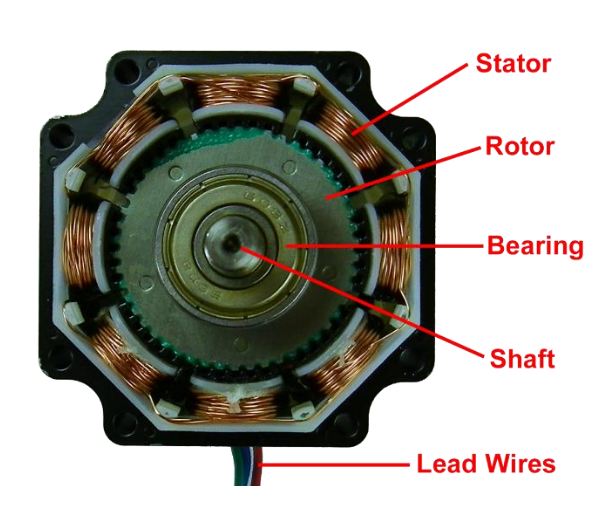


| | | |
|---|---|---|
| A3967 | A4988 | ULN2003 |

في هذا المثال سوف نستخدم المحرك BYJ48-28 كيفية ربطه والتحكم مع Arduino باستخدام وحدة الدرايف ULN2003.

بخلاف محرك DC العادي ، يحتوي هذا المحرك على خمسة أسلاك من جميع الألوان التي تخرج منها ولماذا؟ لفهم هذا يجب علينا أولا أن نعرف كيف يعمل المحرك وما هو تخصصه. أولا وقبل كل المحركات steppers لا تدور! ، فإنها خطوية ولهذا تعرف باسم المحركات الخطوية. بمعنى ، سيتحرك خطوة واحدة فقط في كل مرة. تحتوي هذه المحركات على سلسلة من الملفات الموجودة فيها ، ويجب تنشيط هذه الملفات بطريقة خاصة لجعل المحرك يدور. عندما يتم تنشيط كل ملف ، فإن المحرك يأخذ خطوة وسلسلة من الطاقة سيجعل المحرك يتخذ خطوات مستمرة ، مما يجعله اخيرا يدور. دعونا نلقي نظرة على الملفات الموجودة داخل المحرك لمعرفة بالضبط من أين تأتي هذه الأسلاك.



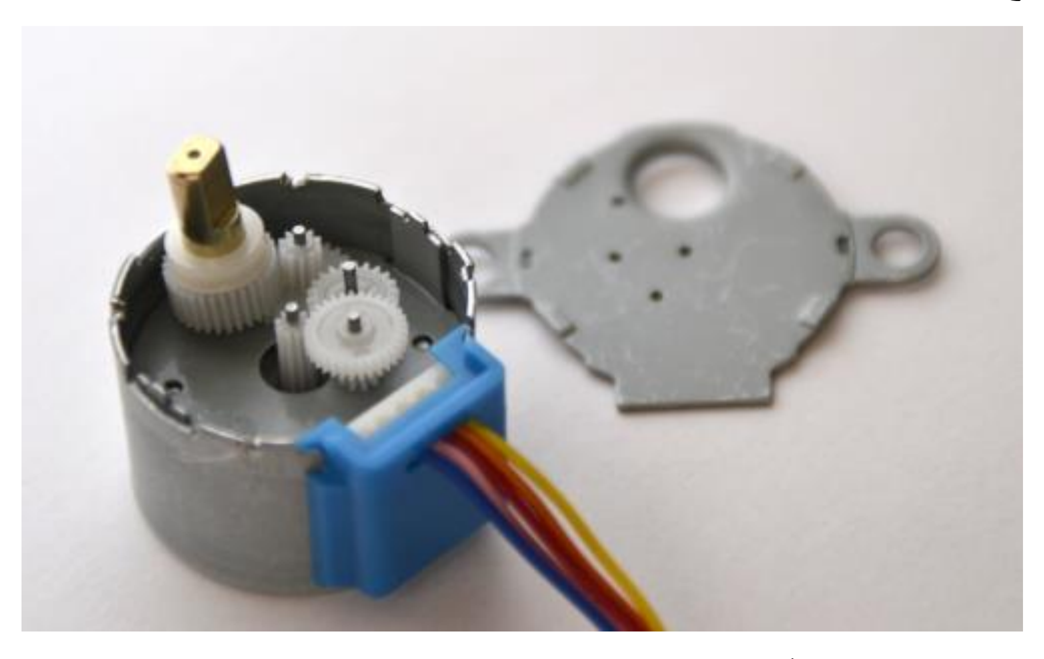
كما ترون المحرك لديه ترتيب من الملفات القطب. هناك أربعة ملفات يجب تنشيطها في تسلسل معين. سيتم تزويد الأسلاك الحمراء بـ + 5 Vوسيتم سحب الأسلاك الأربعة المتبقية إلى الأرض لتشغيل الملف الخاص بها. نستخدم متحكم دقيق مثلا او غيره حيث ينشط هذه الملفات في تسلسل معين ويجعل المحرك يقوم بالعدد المطلوب من الخطوات ليبدأ بالحركة .

دعونا نلقي نظرة على بعض البيانات الفنية الهامة التي تم الحصول عليها من ورقة البيانات الخاصة بهذا المحرك في الصورة أدناه.

| | |
|---|---|
| Rated voltage : | 5VDC |
| Number of Phase | 4 |
| Speed Variation Ratio | 1/64 |
| Stride Angle | 5.625°/64 |
| Frequency | 100Hz |
| DC resistance | 50Ω±7%(25℃) |
| Idle In-traction Frequency | > 600Hz |
| Idle Out-traction Frequency | > 1000Hz |
| In-traction Torque | >34.3mN.m(120Hz) |
| Self-positioning Torque | >34.3mN.m |
| Friction torque | 600-1200 gf.cm |
| Pull in torque | 300 gf.cm |
| Insulated resistance | >10MΩ(500V) |
| Insulated electricity power | 600VAC/1mA/1s |
| Insulation grade | A |
| Rise in Temperature | <40K(120Hz) |
| Noise | <35dB(120Hz,No load,10cm) |
| Model | 28BYJ-48 – 5V |

هذا هي الشيت الكاملة للمعلومات ، ولكننا نحتاج إلى النظر في عدد من المهمات لمعرفة نوع المحرك الخطوي الذي نستخدمه حتى نتمكن من برمجته بكفاءة. أولا نحن نعرف أنه محرك السائر V5 حيث أننا تنشيط السلك الأحمر مع V5. ثم ، نعلم أيضا أنه محرك السائر أربع مراحل لأنه يحتوي على أربعة ملفات فيه. الآن ، يتم تعيين نسبة السرعة إلى 1: 64. هذا يعني أن العمود الذي تراه في الخارج سيجعل دورانًا واحدًا

كاملاً فقط إذا تم تدوير المحرك الداخلي 64 مرة. هذا بسبب التروس التي ترتبط بين المحرك وعمود الإخراج ، تساعد هذه التروس في زيادة عزم الدوران , والشكل التالي يوضح علبة التروس :



ومن البيانات الهامة الأخرى التي يجب ملاحظتها ، زاوية العرض: 5.625 ° / 64. هذا يعني أن المحرك عند العمل في تسلسل من 8 خطوات سيحرك 5.625 درجة لكل خطوة ، وسوف يستغرق 64 خطوة (5.625 * 64 = 360) لإكمال دورة كاملة واحدة.

حساب الخطوات لكل دورة لـ:Stepper Motor

من المهم أن تعرف كيف تحسب الخطوات لكل دورة للمحرك الخطوي لأنه عندها فقط يمكنك برمجة البرنامج بفعالية.

في Arduino سنقوم بتشغيل المحرك في تسلسل من أربع خطوات بحيث تكون زاوية الخطوة هي 11.25 ° حيث أنها 5.625 ° (معطاة في ورقة البيانات) لتسلسل 8 خطوات ستكون 11.25 ° (5.625 * 2 = 11.25).

**خطوات الدورة = 360 / درجة زاوية**

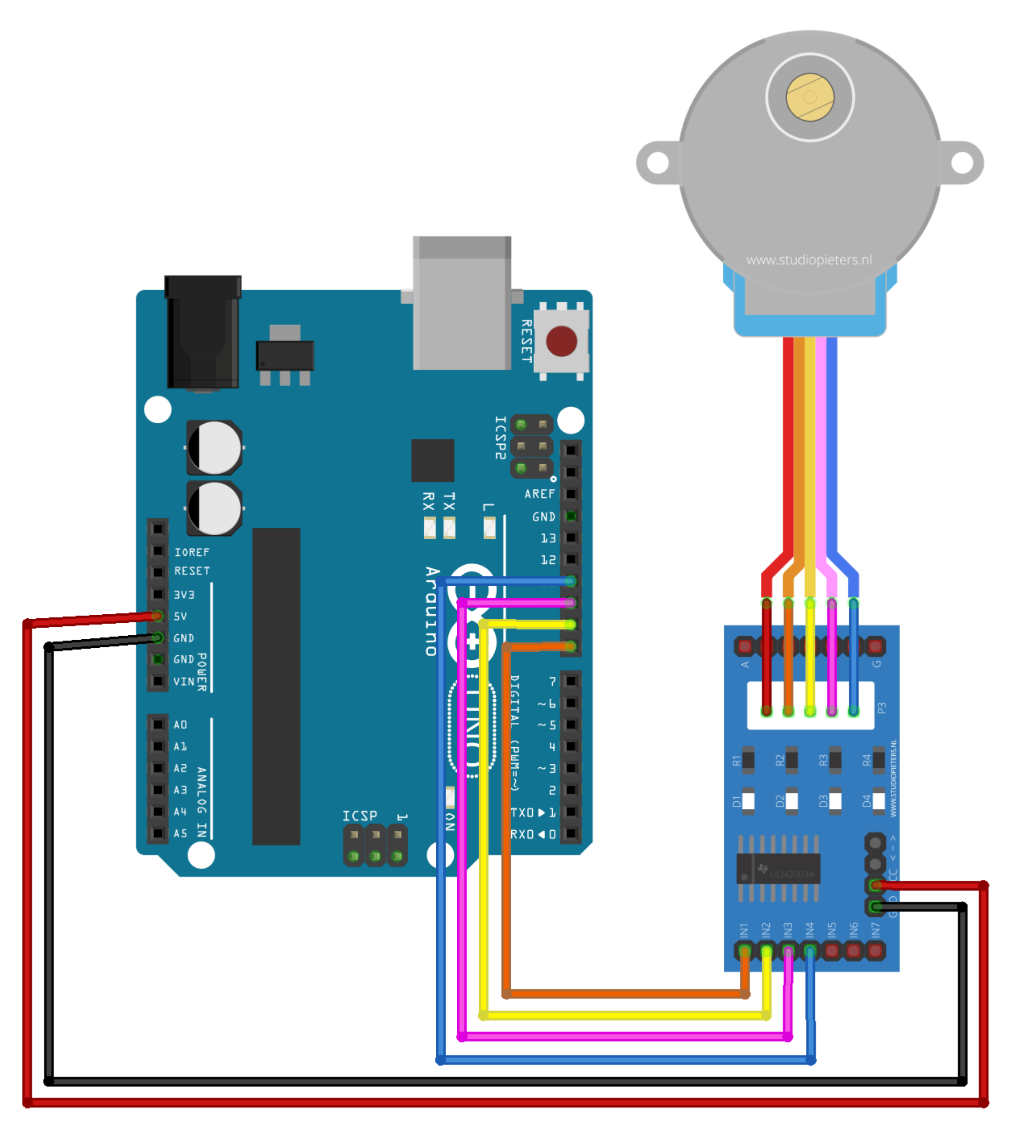
**Steps per revolution = 360/step angle**

هنا ، 360 / 11.25 = 32 خطوة لكل دورة .

لماذا نحتاج إلى مشغل/درايف التشغيل لمحركات السائر(الخطوي)؟

معظم المحركات الخطوية ستعمل فقط بمساعدة وحدة التشغيل. وذلك لأن وحدة التحكم اردوينو لن تكون قادرة على توفير تيار كافي من دبابيس I / O لتشغيل المحرك. لذلك سوف نستخدم وحدة خارجية مثل الوحدة الخارجية ULN2003 هناك العديد من أنواع وحدات التشغيل ، وسيتغير تصنيف اختيارها حسب نوع المحرك المستخدم. سيكون المبدأ الأساسي لجميع وحدات التشغيل هو توفير مصدر التيار بما يكفي لتشغيل المحرك.

مخطط الدائرة :

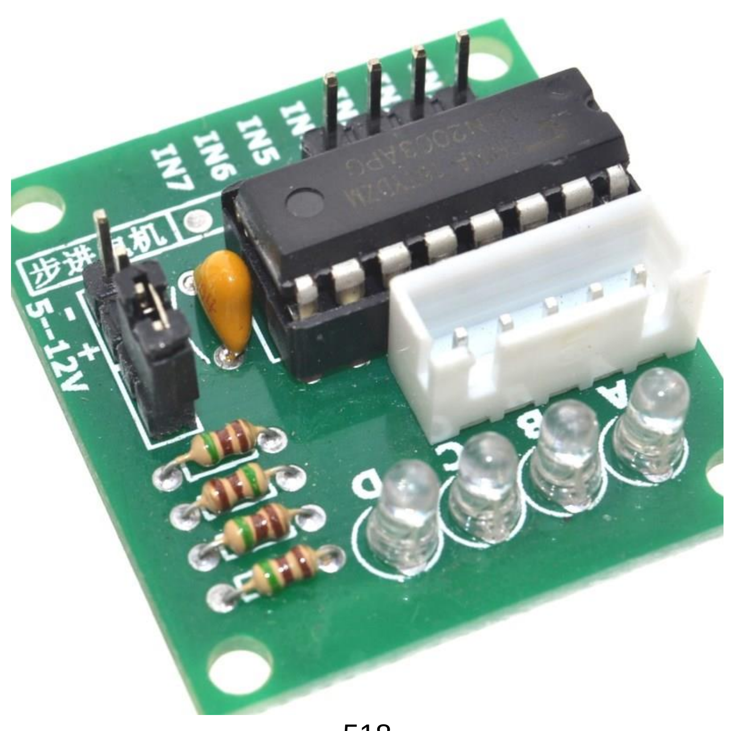


يظهر المخطط اعلاه الدائرة لمشروع التحكم في المحركات السائرة مع arduino لقد استخدمنا محرك السائر 28 BYJ-48وحدة تشغيل .ULN2003 لتنشيط أربع لفات من المحركات السائر نستخدم الدبابيس الرقمية 10،9،8 و 11. يتم تشغيل وحدة السائق عن طريق دبوس 5 Vمن الاردوينو.

قبل أن نبدأ بالبرمجة مع Arduino ، دعونا نفهم ما يجب أن يحدث بالفعل داخل البرنامج. وكما ذكرنا سابقًا ، سنستخدم طريقة تسلسلية من 4 خطوات ، لذا سيكون لدينا أربع خطوات يمكن تنفيذها لإجراء تدوير كامل.

| Step | Pin Energized | Coils Energized |
|---|---|---|
| Step 1 | 8 and 9 | A and B |
| Step 2 | 9 and 10 | B and C |
| Step 3 | 10 and 11 | C and D |
| Step 4 | 11 and 8 | D and A |

ان وحدة التشغيل تحتوي وحدة التشغيل على أربعة مصابيح LED بتسمية ( A B C D) يمكن من خلالها التحقق من تنشيط اي الملف في أي وقت.



سنقوم في هذا البرنامج بكتابة ببرمجة Arduino بطريقة يمكننا من خلالها إدخال عدد الخطوات التي يتعين على محرك السائر اتخاذها من خلال الشاشة التسلسلية serial monitor لـ Arduino .

تم حساب عدد الخطوات لكل ثورة لمحرك السائر لدينا ليكون 32 ؛ ومن ثم ندخل ذلك كما هو موضح في السطر أدناه

```
#define STEPS 32
```

بعد ذلك ، عليك إنشاء حالات نحدد فيها المسامير التي ربطنا بها المحرك

```
Stepper stepper (STEPS, 8, 10, 9, 11);
```

بما أننا نستخدم مكتبة Arduino stepper ، فيمكننا ضبط سرعة المحرك باستخدام الكود أدناه. يمكن أن تتراوح السرعة بين 0 إلى 200 للمحركات السائر BYJ48-28.

```
stepper.setSpeed(200);
```

الآن ، لجعل المحرك يتحرك خطوة واحدة يمكننا استخدام السطر التالي.

```
stepper.step(val);
```

سيتم توفير عدد الخطوات المطلوب نقلها من خلال متغير "val". نظرًا لأن لدينا 32 خطوة و 64 مثل نسبة التروس ، نحتاج إلى نقل 2048 (32 * 64 = 2048) ، لإجراء تدوير كامل واحد.

ويمكن إدخال قيمة المتغير "val" من قبل المستخدم باستخدام الشاشة التسلسلية.

الكود :

```
// Arduino stepper motor control code

#include <Stepper.h> // Include the header file

// change this to the number of steps on your motor

#define STEPS 32

// create an instance of the stepper class using the steps and pins

Stepper stepper(STEPS, 8, 10, 9, 11);

int val = 0;


void setup() {

  Serial.begin(9600);

  stepper.setSpeed(200);

}


void loop() {

  if (Serial.available()>0)

  {

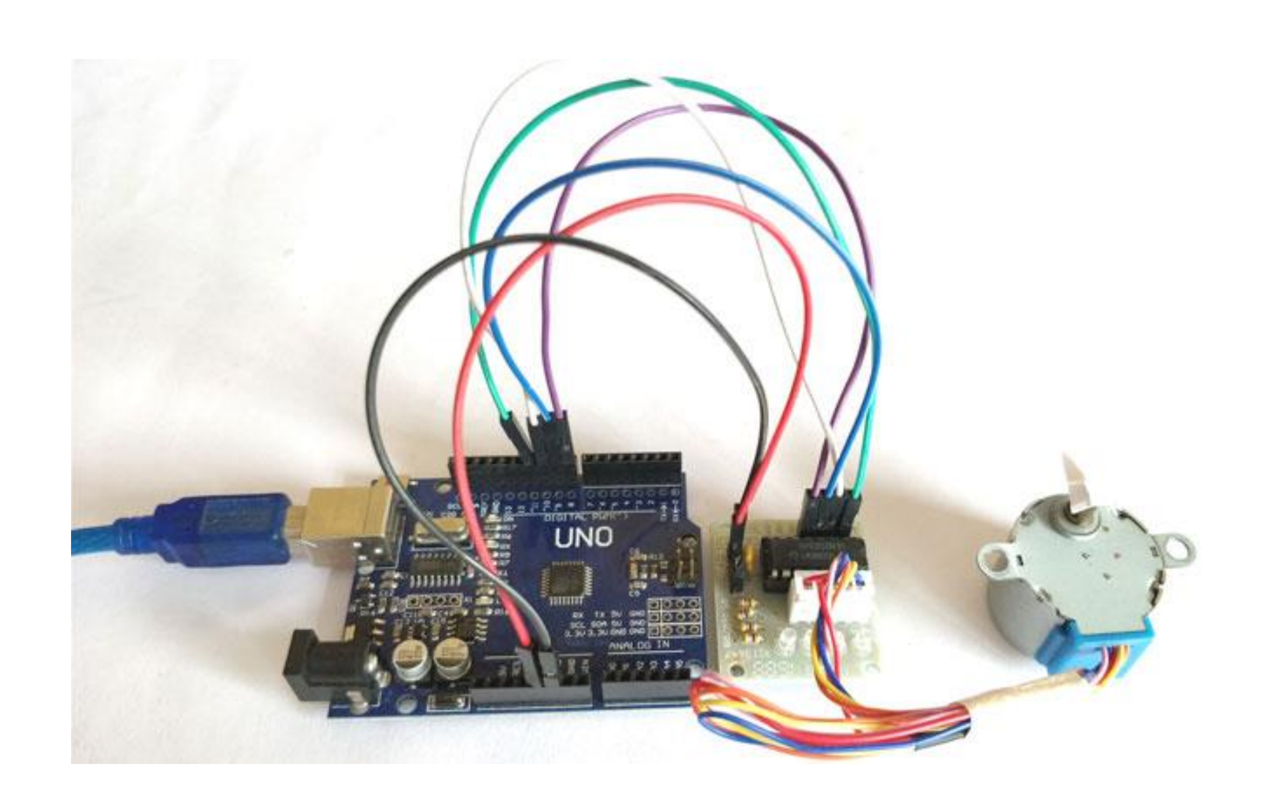
    val = Serial.parseInt();

    stepper.step(val);

    Serial.println(val); //for debugging

  }

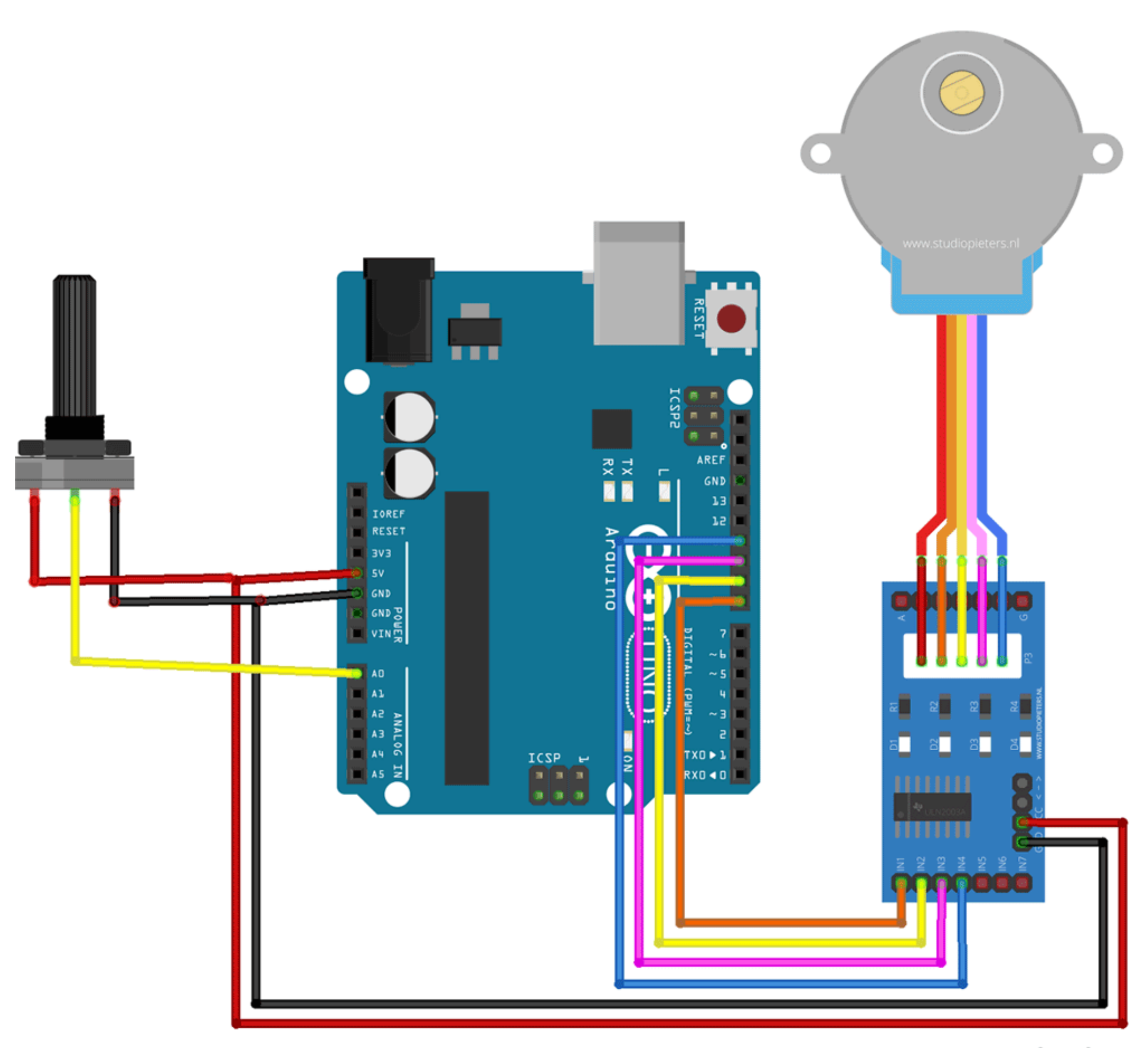
}
```




الآن وبعد تحميل البرنامج افتح الشاشة التسلسلية. كما ناقشنا سابقا سيكون علينا أن نجعل 2048 خطوة لجعل دوران كامل ، لذلك عندما ندخل عام 2048 ، سوف يقوم المحرك بتدوير كامل في اتجاه عقارب الساعة من خلال اتخاذ 2048 خطوة. للتدوير بعكس عقارب الساعة فقط أدخل الرقم مع "-" علامة سالبة. لذا ، فإن دخول -1024 سيجعل المحرك يدور في منتصف الطريق في الاتجاه المعاكس للساعة. يمكنك إدخال أي قيم مطلوبة ، مثل إدخال( will 1) تجعل المحرك يأخذ خطوة واحدة فقط.

- التحكم في المحرك الخطوي يدويا باستخدام مجزء الجهد

المخطط :



الكود :

```
#include <Stepper.h> // Include the header file



// change this to the number of steps on your motor

#define STEPS 32



// create an instance of the stepper class using the steps and pins

Stepper stepper(STEPS, 8, 10, 9, 11);



int Pval = 0;

int potVal = 0;



void setup() {

  Serial.begin(9600);

  stepper.setSpeed(200);

}



void loop() {



potVal = map(analogRead(A0),0,1024,0,500);

if (potVal>Pval)

  stepper.step(5);
```

```
if (potVal<Pval)

  stepper.step(-5);

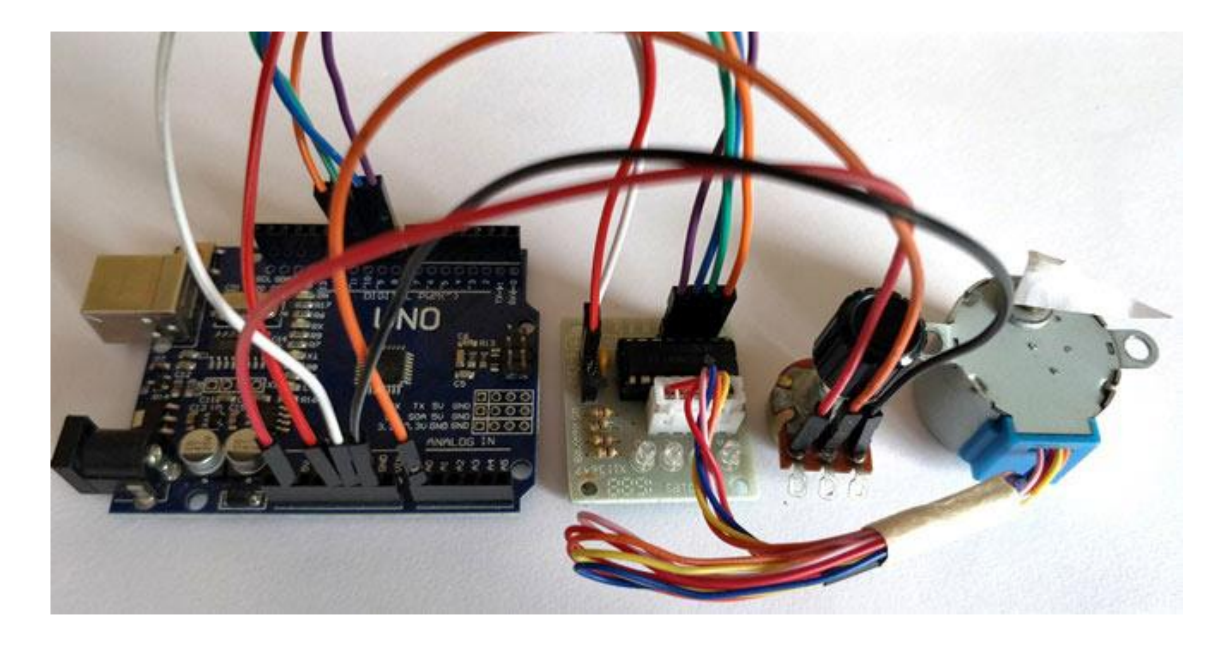


Pval = potVal;

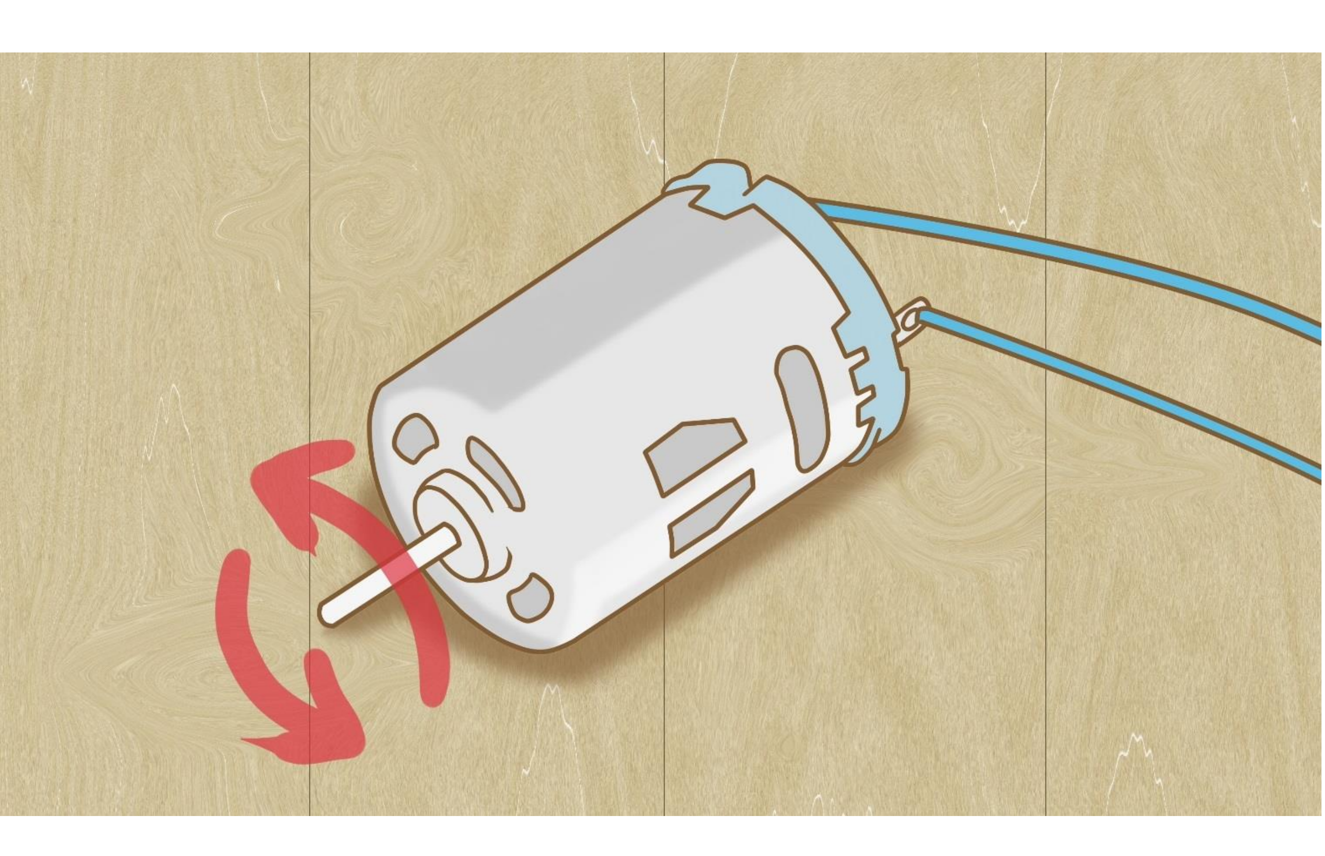


Serial.println(Pval); //for debugging

}
```



- ثالثا : التعامل مع محرك التيار المباشر **DC**



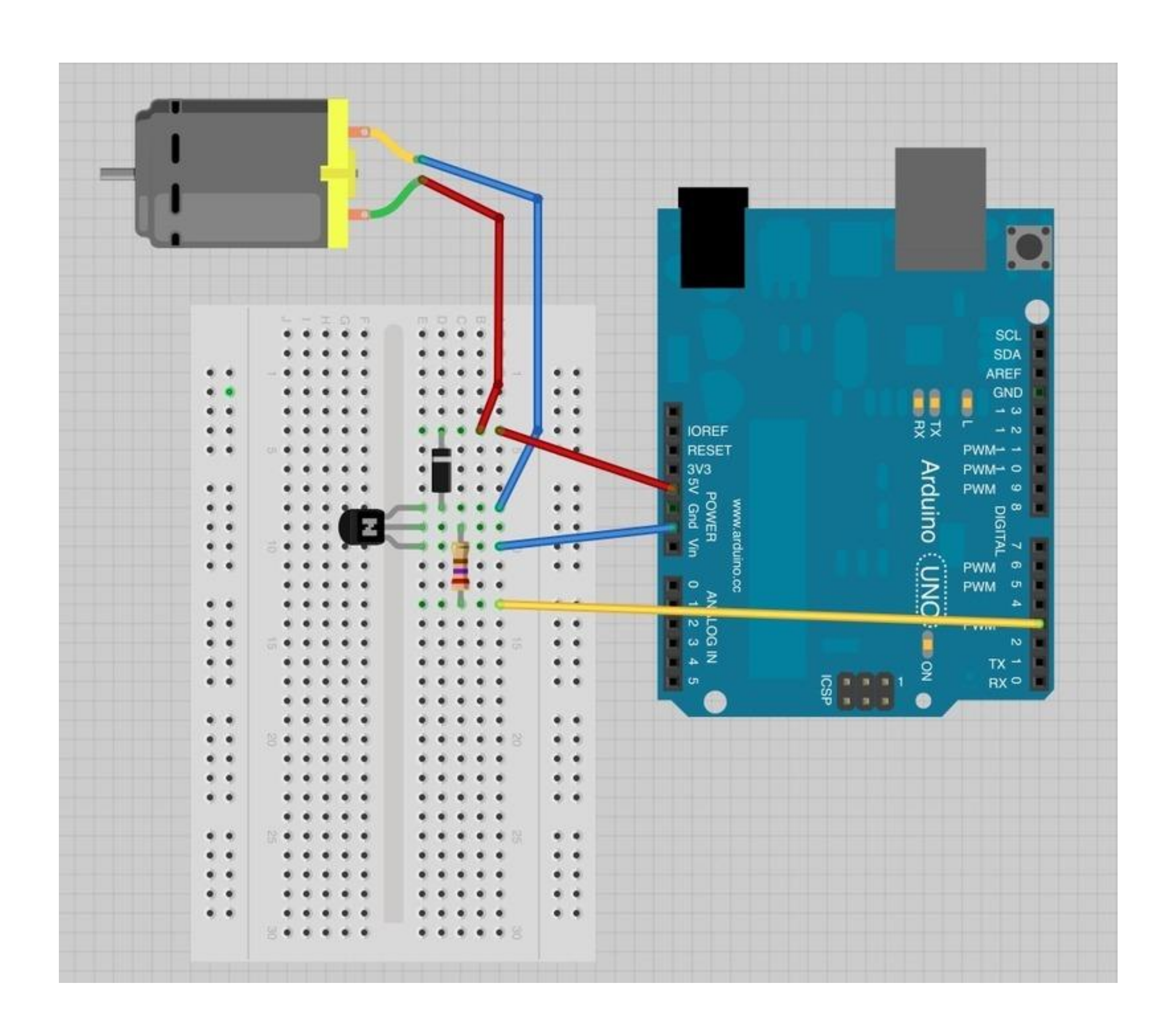
محرك التيار المستمر (محرك DC) هو محرك كهربائي يحول الطاقة الكهربائية إلى طاقة حركية ويعمل فقط على أنظمة التيار المستمر، تعتبر المحركات الكهربائية القوة المحركة لكثير من التطبيقات الصناعية. وتستهلك المحركات الكهربائية بأنواعها المختلفة حوالي 60% من الطاقة الكهربائية في العالم , وتعد محركات التيار المستمر من أهم الانواع حيث تستخدم في الجر الكهربائي والروافع وصناعة الغزل والنسيج ودرفلة الحديد وفي صناعة الورق والاسمنت والسيارات والطائرات والقطارات , وذلك لما تتميز به من سهولة التحكم في سرعتها وإعطائها عزم مرتفع وخصوصا عند بدء الحركة.

في هذا المثال ستتعلم كيف تتحكم بمحرك التيار المستمر الصغير DC Motor ستقوم بإستخدام خاصية تغيير المخرج التناظري ‘ – Pulse Width Modulation pwm’ بالأردوينو وذلك للتحكم في سرعة المحرك عبر ارسال رقم مابين 0 و 255 من شاشة الاتصال التسلسلي Serial Monitor.

سوف نحتاج الى القطع التالية :

| | |
|---|---|
| | 6V DC Motor |
| | PN2222 Transistor |
| | 1N4001 diode |
| | 270Ω Resistor |

عند وضع القطع على لوح التجارب عليك ان تحرص على صحة اتجاه وموضع الترانزستور والصمام الثنائي 'diode' كما في الصورة.



الكود:

```
int motorPin = 9;


void setup() {

   pinMode(motorPin, OUTPUT);
```

```
    Serial.begin(9600);

    while (! Serial);

    Serial.println("Speed 0 to 255");

}


void loop() {

    if (Serial.available()) {

        int speed = Serial.parseInt();

        if (speed >= 0 && speed <= 255) {

            analogWrite(motorPin, speed);

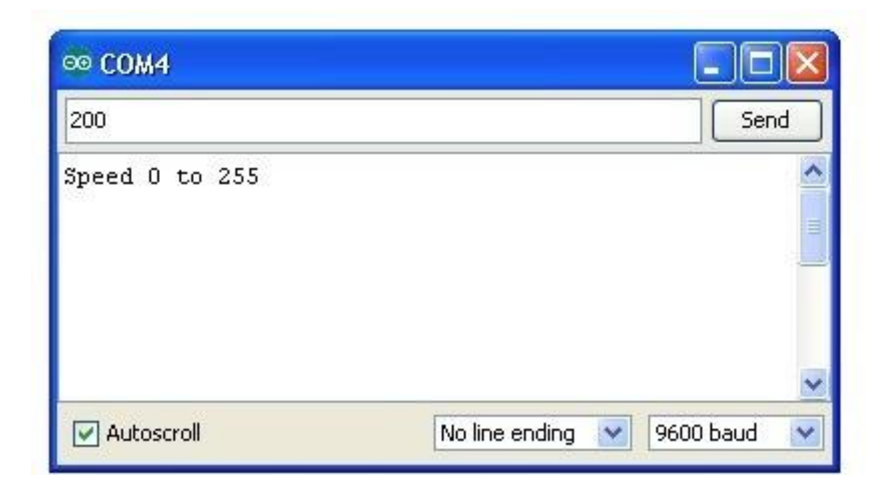
        }

    }

}
```

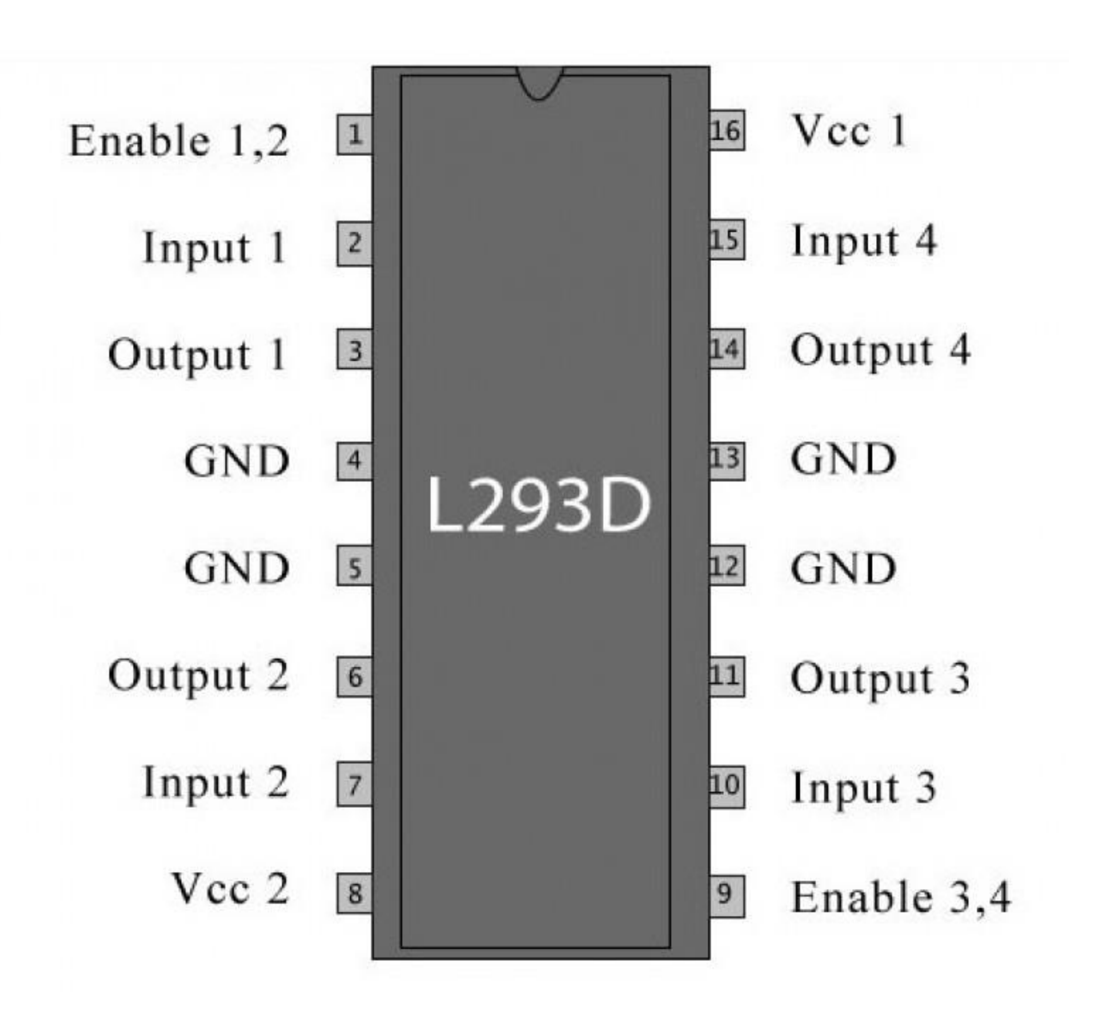
يؤدي الترانزستور هنا وظيفة المحول 'switch' وذلك عبر التحكم بطاقة المحرك. نقوم بإستخدام منفذ 3 للأردوينو لتشغيل وإطفاء الترانزستور تحت اسم 'motorPin'.

عند بدء البرنامج ستظهر شاشة الاتصال التسلسلي Serial Monitor تطلب منك ادخال قيمة طاقة المحرك ( مابين 0 و 255 ).



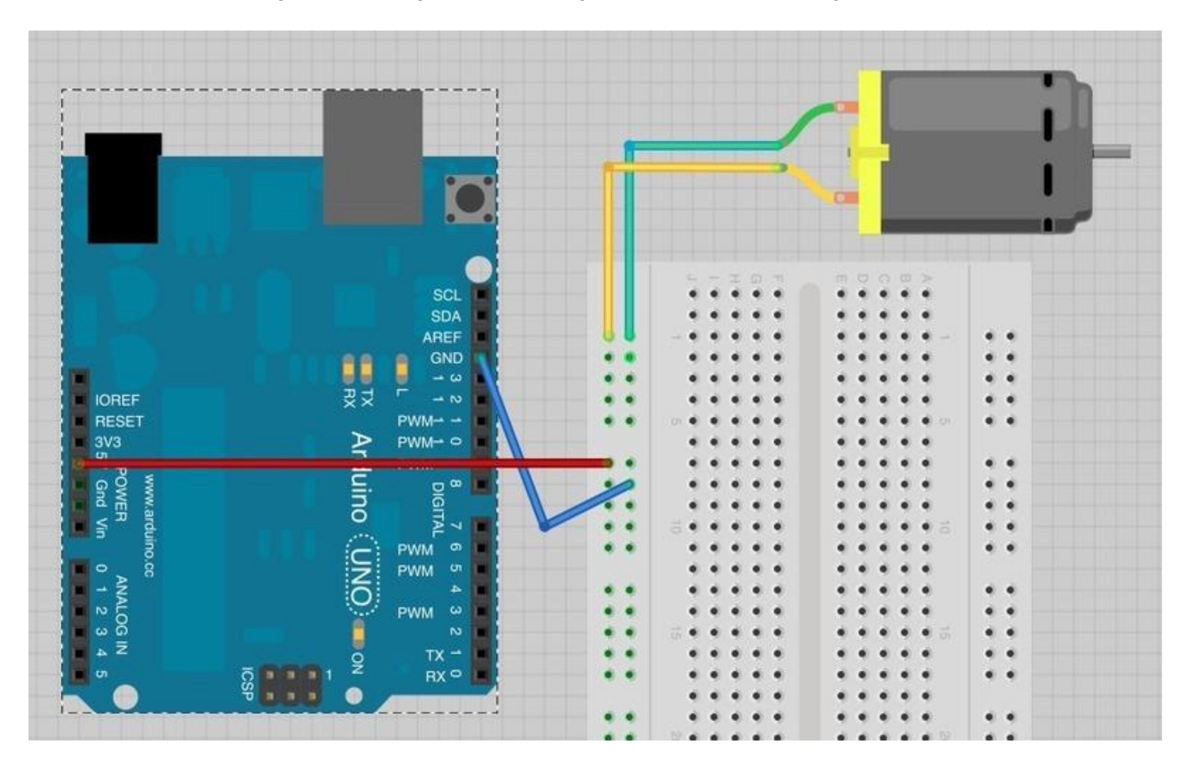
سنتعلم الآن كيفية التحكم بسرعة واتجاه حركة محرك التيار المستمر الصغير DC Motorعن طريق الأردوينو ورقاقة التحكم بالمحرك L293D :

في هذه التجربة استخدمنا المقاوم المتغيرة 'pot' للتحكم بسرعة المحرك، كما استخدمنا زر 'push button' للتحكم في اتجاه حركة المحرك , ونحتاج الى L293D IC الموضحة في الشكل التالي :




قبل ان تحكم بالمحرك الصغير ، علينا اجراء التجربة مع رقاقة التحكم بالمحرك L293Dللتعرف على طريقة عملها.

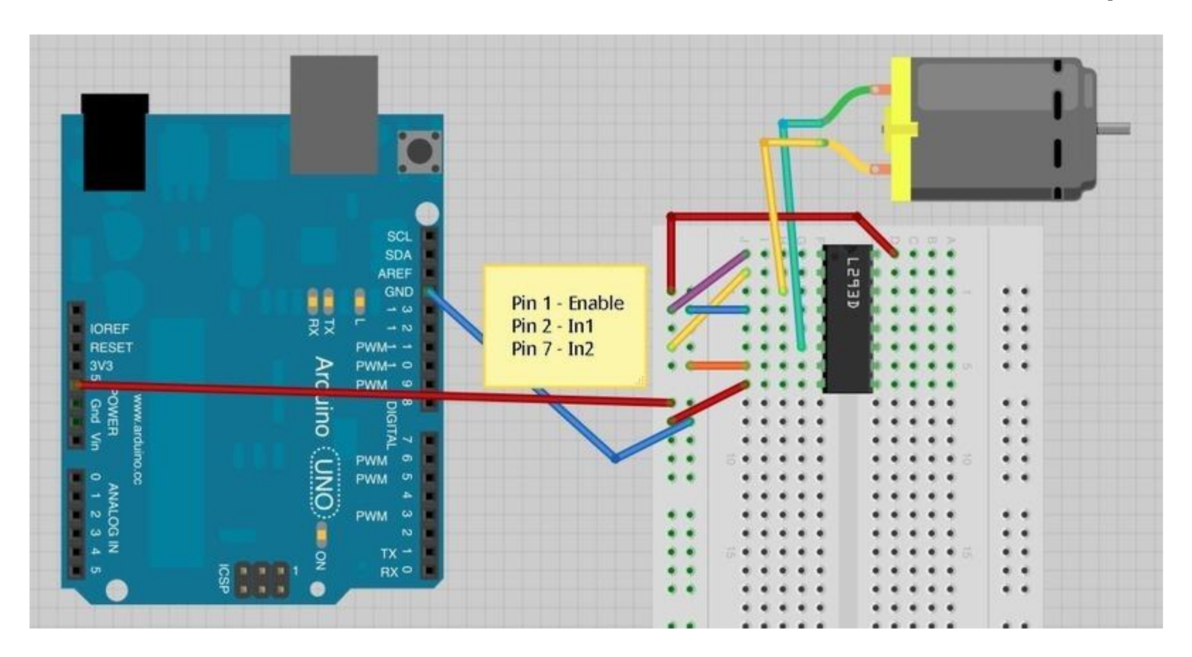
كل ماعلينا فعله هو توفير طاقة 5 Vمن الأردوينو للمحرك الصغير.




لاحظ اتجاه دوران المحرك ، يمكنك لمس المحرك بأصابعك لمعرفة اتجها الحركه . يمكنك عكس اتجاه حركة المحرك عبر عكس توصيلات المحرك (قم بتبديل توصيلة الـ5 Vبـ GND، كذلك الحال مع التوصيله الأخرى) بذلك سترى انعكاس لاتجاه حركة الدوران.

وهذا هو دور رقاقة التحكم بالمحركL293D ، تقوم بالتحكم بالمنافذ التي توفر الطاقة والمجال الأرضي GND وبذلك تتحكم باتجاه حركة الدوران.

قم ببناء لوح التجارب كالتالي ، لاحظ ان دور متحكم الأردوينو هنا هو توفير الطاقه فقط.




المنافذ التي تهمنا بالنسبة لرقاقة التحكم بالمحرك L293D هي ثلاث منافذ ، منفذ 1 Pin 1 (Enable)، منفذ 2 Pin 2 (In1) ، ومنفذ 7. (In2) Pin 7

وهذه تكن موصولة إما بـ5 Vأو GND باستخدام السلك البنفسجي ، أو الأصفر أو البرتقالي.

اذا قمت بربط Pin 1 (Enable) للـ GNDسيتوقف المحرك، مهما فعلت بـ pin In1و pin In2 يقوم Enable بالتشغيل أو الإطفاء. هذا يجعله مفيد لاستخدام 'PWM output'للتحكم في سرعة المحرك. قم بتوصيل Pin 1 إلى 5 Vلتشغيل المحرك مره أخرى.

والان قم بتغيير pin 2,) In1 أصفر) من 5 Vإلى.GND

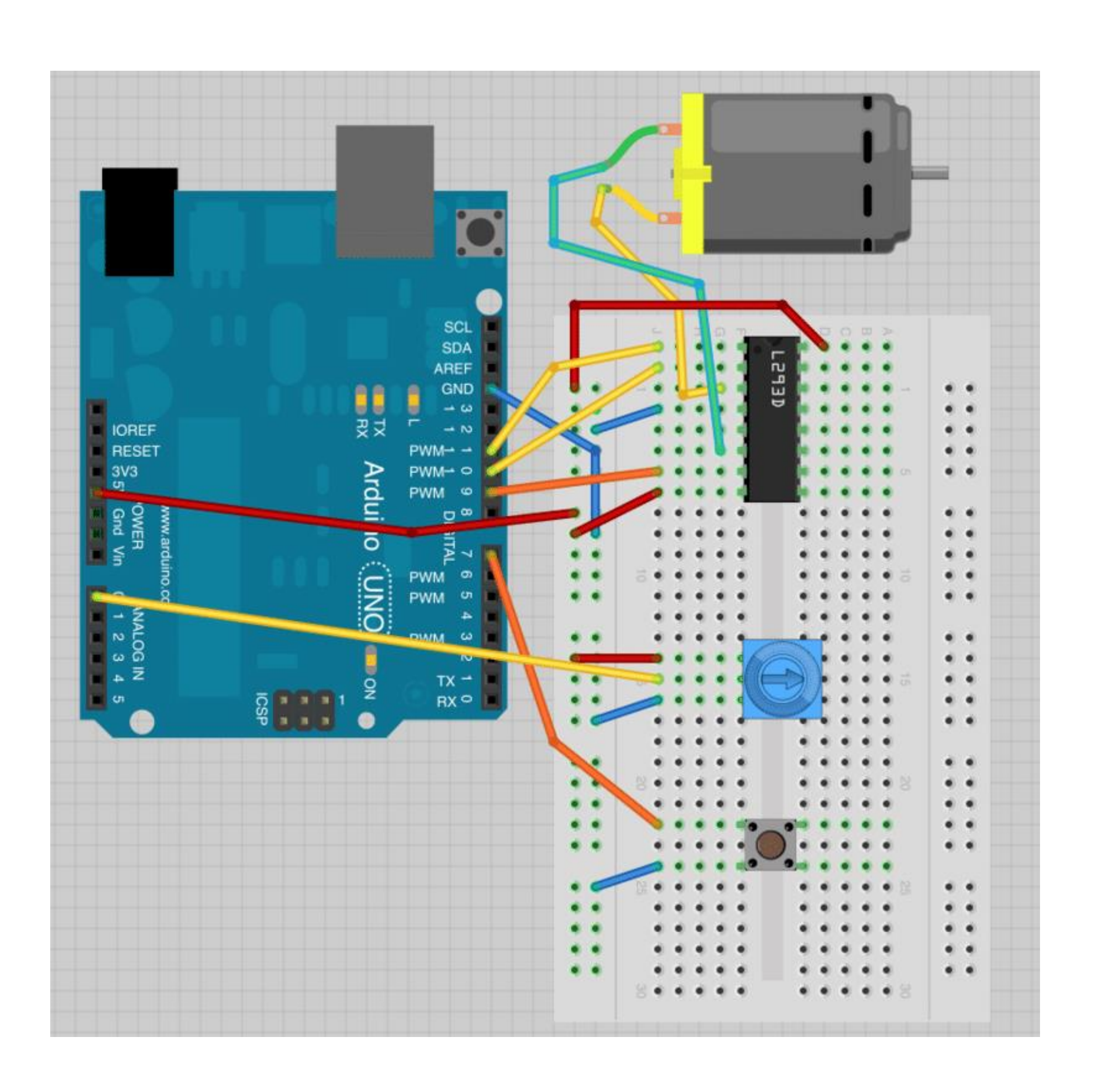
In1و In2 جميعها مرتبطه بـ GND، وبالتالي سيتوقف المحرك مره أخرى.

تغيير In2 من GND إلى 5 Vسيعكس حركة دوران المحرك ( الاتجاه الآخر ).

أخيراً ، بتغيير In1 وربطه مره أخرى بـ 5 Vبالتالي سيكون In1 و In2 مرتبطه بـ5 Vسيوقف المحرك.

والآن وبعد ان اعتدت على التحكم بالمحرك ورحة دورانه ، يمكننا الآن جعل متحكم الأردوينو يتحكم بمنافذ Enable, In1, In2

عند بناء لوح التجارب عليك التأكد من موضع واتجاه الرقاقة كما هي في التصميم (الطرف الذي حيتوي على نتوء يجب أن يكون باتجاه الأعلى )



قم برفع الكود التالي على متحكم الأردوينو :

```
int enablePin = 11;

int in1Pin = 10;

int in2Pin = 9;

int switchPin = 7;

int potPin = 0;


void setup()

{

  pinMode(in1Pin, OUTPUT);

  pinMode(in2Pin, OUTPUT);

  pinMode(enablePin, OUTPUT);

  pinMode(switchPin, INPUT_PULLUP);

}

void loop()

{

  int speed = analogRead(potPin) / 4;

  boolean reverse = digitalRead(switchPin);

  setMotor(speed, reverse);

}

void setMotor(int speed, boolean reverse)

{

  analogWrite(enablePin, speed);

  digitalWrite(in1Pin, ! reverse);

  digitalWrite(in2Pin, reverse);

}
```

داخل دالة setup تم تعريف المنافذ وحالاتها.

داخل دالة loop ، يتم تحديد قيمة السرعة للمحرك عبر اخذ قراءة القيمة التناظرية 'analogRead' من المقاوم المتغير 'pot' وقسمته على 4.

السبب وراء القسمة على 4 هي لأن القيمة المستخرجة من القراءة التناظرية 'analogRead' تكون مابين 0 و 1023 ولكنها يجب أن تكون القيمة بين 0 و 255.

اذا تم الضغط على الزر ، فإن المحرك سيتحرك للأمام ، وبالضغط مره أخرى سيتحرك بعكس الاتجاه.

القيمة لمتغير 'reverse' يتم أخذه من متغير 'switchPin'، لذا عند الضغط على الزر ستكون القيمة 0،وعند الضغط مره أخى ستكون القيمة 1.

قيمة السرعة و الانعكاس 'reverse' يتم تمريرها إلى الدالة 'setMotor' والتي ستحدد المنافذ لرقاقة التحكم بالمحرك للتحكم .

```
void setMotor(int speed, boolean reverse)

{

  analogWrite(enablePin, speed);

  digitalWrite(in1Pin, ! reverse);

  digitalWrite(in2Pin, reverse);

}
```

أولاً ، السرعة تم تحديدها عبر استخدام 'analogWrite' لمنفذ enable pin . منفذ enable pin يقوم بتشغيل أو اطفاء المحرك بغض النظر عن قيم منافذ in1 و in2.

للتحكم باتجاه حركة الدوران للمحرك علينا عكس قيم منافذ in1 وin2

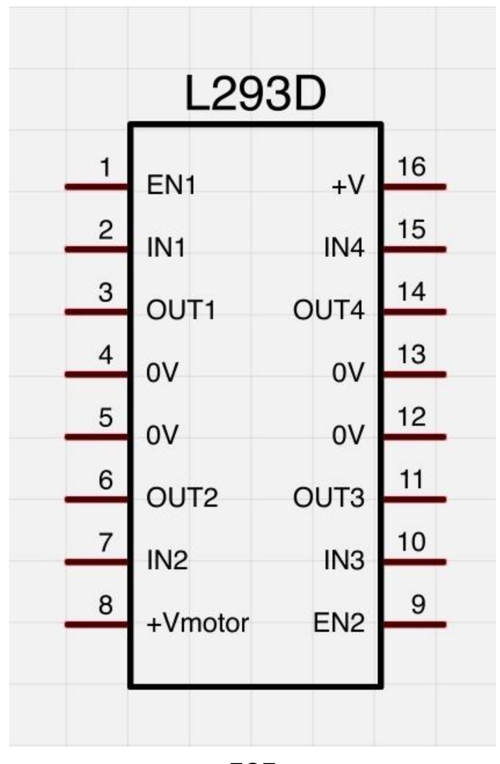
اذا كانت قيمة in1 عالية (اي تساوي 1) و قيمة in2 منخفضة (اي تساوي 0)، سيقوم المحرك بالدوران باتجاه واحد ، واذا تم عكس القيم لمنافذ in1 و in2 فسيقوم المحرك بالدوران بالاتجاه المعاكس.

الأمر ‘ ! ‘ يعني ‘ ليس 'not – لذلك أول امر ‘digitalWrite’ لـ in1يقوم بعكس القيمة المأخوذه من ‘reverse’ فمثلاً لو كانت القيمة عالية (تساوي 1) فسيوقم بعكسها إلى قيمة منخفضة (تساوي 0) والعكس صحيح.

الأمر الثاني ‘digitalWrite’ لـ in2يقوم بأخذ القيمة من ‘reverse’ وهذا يعني بأنها ستكون دائما عكس القيمة لـ in1

## رقاقة التحكم بالمحركL293D

هذه الرقاقة مفيده جداً ، علماً بأنها تستطيع التحكم على محركين اثنين بنفس الوقت وبشكل منفصل . نقوم بإستخدام نصف الرقاقة بهذا الدرس ، معظم المنافذ الموجوده على يمين الرقاقة هي للتحكم بالمحرك الثاني.



المحرك الثاني يمكن ربطه بين OUT3 و .OUT4 كما يجب توفير ثلاث منافذ تحكم.

EN2متصل بـ ’PWM enabled output‘ بالأردوينو.

IN3متصل بالمخارج الرقمية ‘digital outpus’ بالأردوينو

الرقاقة L293D تمتلك منفذين طاقة V+ ( منفذ 8 و منفذ 16). منفذ 8 يوفر الطاقة للمحركات ، ومنفذ 16 يوفر الطاقة للرقاقة. قمنا بربطها جميعاً بمنفذ 5 Vبالأردوينو.

عموماً اذا كان لديك محرك يحتاج طاقة أعلى من ذلك فعليك توفير الطاقة بشكل منفصل للمحرك باستخدام المنفذ 8.

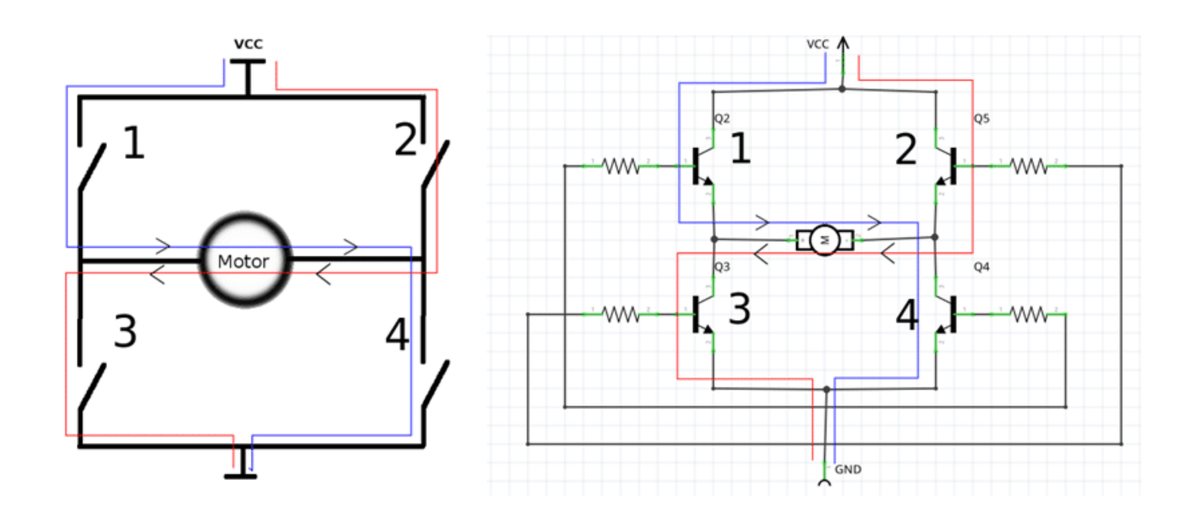
والآن تستطيع تغيير الكود البرمجي بحيث يمكنك التحكم بالمحر دون الحاجة للمقاوم المتغير و الزر . بحيث تبدأ حركة المحرك بشكل بطيء باتجاه واحد ، ثم تزداد السرعة بشكل تدريجي ، ثم تبطئ مره أخرى ويتم عكس الاتجاه وهكذا..

## التحكم في محرك تيار مستمر باستخدامH-Bridge

في هذا المشروع سنتعلم فكرة عمل ال H-Bridgeوكيفية استعماله للتحكم في تشغيل وايقاف محرك تيار مستمر وايضا عكس اتجاه حركته. قد يستخدم في روبوت متتبع الخط او اي روبوت نحتاج للتحكم في اتجاه حركته.

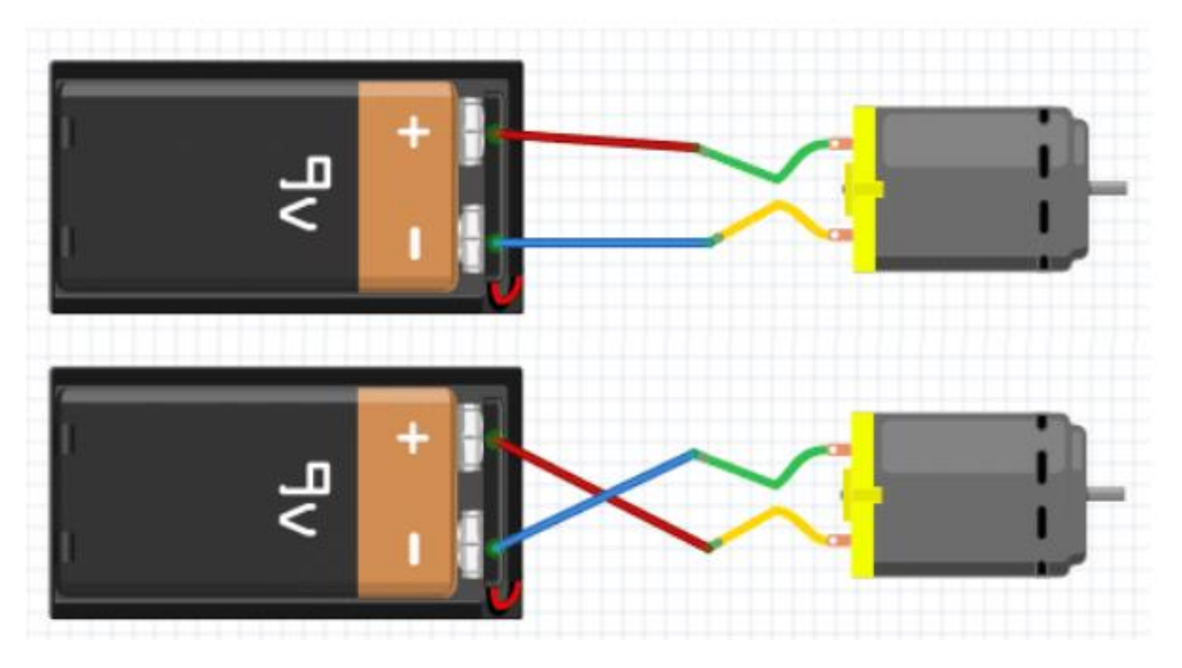
H-Bridge هو عبارة عن أربعة ترانزستور موصلين معا بشكل معين لتمكين المحرك من الدوران في إتجاهين مختلفين.

فكرة عمله , الصورة التالية توضح كيف يتركب الـ H-Bridge:




عند تشغيل كلا الترانزستور 1 و 4 يعمل المحرك نحو الإتجاه الأول وعند تشغيل الترانزستور 2 و 3 يعمل المحرك في الإتجاه المعاكس للإتجاه السابق.

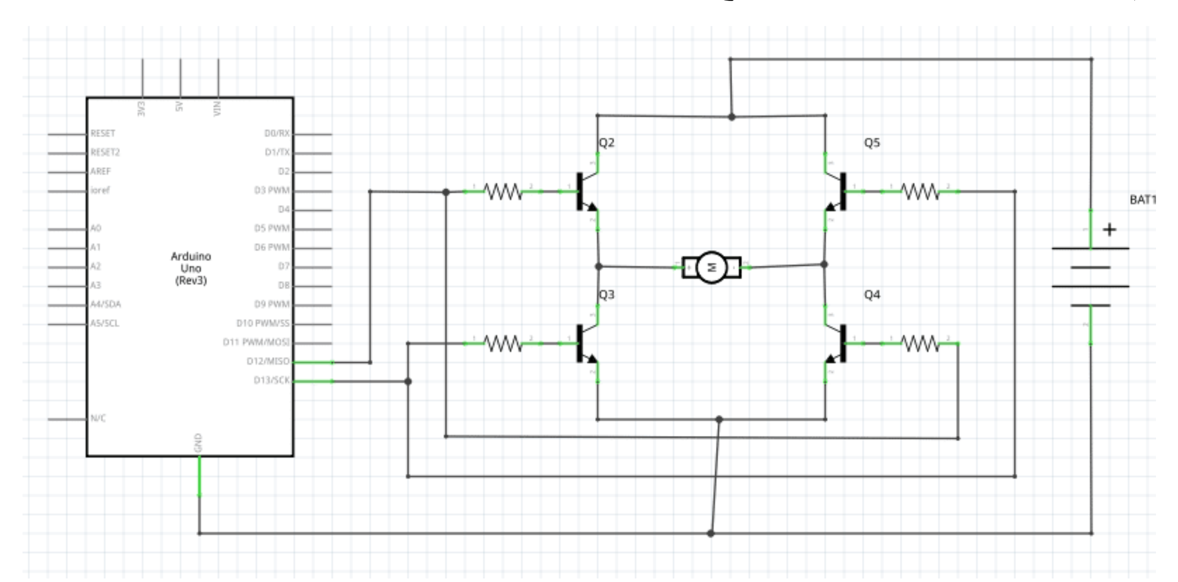
وبذلك، يتم عكس الدوران اتوماتيكيا دون الحاجة إلى تغيير التوصيل يدويا. على عكس محرك التيار المستمر، الذي يتم تعكس اتجاهه عن طريق عكس اطراف المحرك مع مصدر التيار الكهربائي.



في الحالة الأولى، سيدور المحرك مع إتجاه عقارب الساعة. وبعكس توصيل الأطراف (كما هو موضح بالحالة الثانية) سينعكس إتجاه الدوران ليصبح عكس عقارب الساعة.

لا يمكن توصيل محرك التيار المستمر مباشرة مع الأردوينو. وذلك لأن المحرك يحتاج إلى تيار عالي لا يستطيع الأردوينو إعطاءه له. لذلك، سنقوم بإستخدام الترانزستور كدائرة بين الاردوينو الذي يعمل مع تيار صغيرة وبين المحرك الذي يحتاج إلى تيار عالي.

قم بتوصيل الدارة كما هو موضح بالصورة:




ولنتمكن من تشغيل المحرك في إتجاهين، يتم تشغيل الطرف الأول من الـ H-Bridge للأردوينو في المحرك، فيدور المحرك في الإتجاه الأول. وعند تشغيل الطرف الثاني سيدور بالإتجاه المعاكس.

في هذا المشروع، سيدور المحرك في الإتجاه الأول لمدة ثلاث ثوان، ثم يتوقف لمدة ثلاث ثوان. ثم يدور المحرك بالإتجاه المعاكس لمدرة ثلاث ثوان، ثم يتوقف لمدة ثلاث ثوان أخرى. وهكذا حتى يتم فصل التيار الكهربائي , وقم قم بتحميل الكود التالي إلى الأردوينو :

```
#define MOTOR_IN1 12

#define MOTOR_IN2 13



void motor_forward(void);  // a function that will be called to rotate it
clockwise
```

```
void motor_reverse(void);  // a function that will be called to totate it counter-clockwise

void motor_stop(void);     // a function that will be called to stop the rotation



void setup() {

  pinMode(MOTOR_IN1, OUTPUT);  // set the first pin of the relay as output

  pinMode(MOTOR_IN2, OUTPUT);  // set the 2nd pin of the relay as output

}



void loop() {

  motor_forward();             // move forward/clockwise

  delay(3000);                 // keep rotating cw for 3 seconds

  motor_stop();                // stop rotating

  delay(3000);                 // stand still for 3 seconds

  motor_reverse();             // reverse the rotation direction/ccw

  delay(3000);                 // keep rotating ccw for 3 seconds

  motor_stop();                // stop rotating

  delay(3000);                 // stand still for 3 seconds

}



void motor_forward(void)       // the function that will cause the motor to rotate cw

{

  digitalWrite(MOTOR_IN1, HIGH);

  digitalWrite(MOTOR_IN2, LOW);

}
```

```
void motor_reverse(void)       // the function that will cause the motor to rotate ccw

{

  digitalWrite(MOTOR_IN1, LOW);

  digitalWrite(MOTOR_IN2, HIGH);

}



void motor_stop(void)          // the function that will cause the motor to stop rotating

{

  digitalWrite(MOTOR_IN1, LOW);

  digitalWrite(MOTOR_IN2, LOW);

}
```

شرح الكود :

قمنا سابقا بتوصيل طرفى كلا من الترانزستور (IN1,IN2) بمنفذ 12 و 13 للأردوينو لذلك قمنا بتسمية كلا المنفذين للأردونو تبعا لما تم توصيله بالدارة.

```
#define MOTOR_IN1 12

#define MOTOR_IN2 13
```

نقوم بتعرف المتغيرات IN1 و ( IN2 أطراف الـ H-bridge الموصله بالاردوينو) كمخرج.

```
void setup() {

  pinMode(MOTOR_IN1, OUTPUT);  // set the first pin of the relay as output

  pinMode(MOTOR_IN2, OUTPUT);  // set the 2nd pin of the relay as output

}
```

في دالةloop() ، نقوم أولا بإستدعاء الدالة . motor_forward() تقوم هذه الدالة بتشغيل المحرك مع اتجاه عقارب الساعة لمدة 3 ثوان . (delay(3000)) ثم نقوم

باستخدام الدالة motor_stop() ، لإيقاف المحرك عن العمل لمدة 3 ثوان. ثم يتم عكس اتجاه حركة المحرك باستخدام الدالة motor_reverse() لمدة 3 ثوان. ومن ثم يعود ليكرر نفس هذه المهمة من البداية مرة أخرى.

```
void loop() {

  motor_forward();            // move forward/clockwise

  delay(3000);                // keep rotating cw for 3 seconds

  motor_stop();               // stop rotating

  delay(3000);                // stand still for 3 seconds

  motor_reverse();            // reverse the rotation direction/ccw

  delay(3000);                // keep rotating ccw for 3 seconds

  motor_stop();               // stop rotating

  delay(3000);                // stand still for 3 seconds

}
```

الدلةmotor_forward() ، تقوم بتحريك المحرك بإتجاه عقارب الساعة. تتم هذه العملية عن طريق جعل قيمة IN1 للمرحل HIGH والطرف الآخر. LOW

```
void motor_forward(void)      // the function that will cause the motor to
rotate cw

{

  digitalWrite(MOTOR_IN1, HIGH);

  digitalWrite(MOTOR_IN2, LOW);

}
```

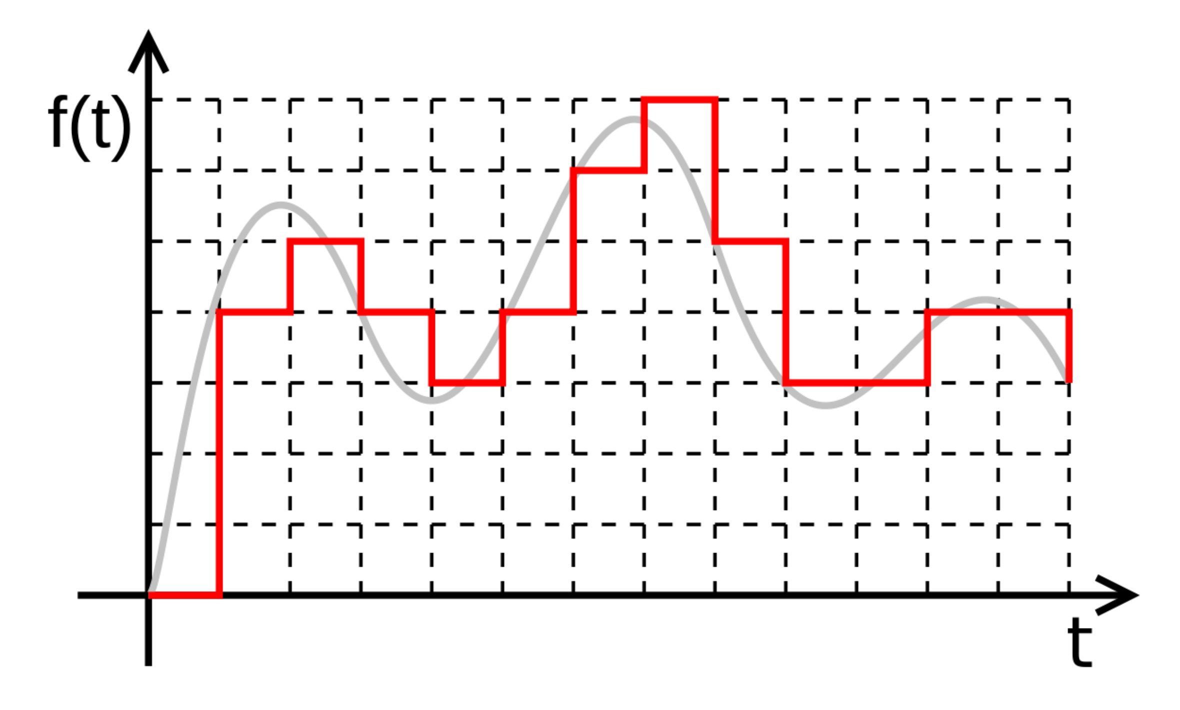
تعمل هذه الدالة motor_reverse() بشكل مشابه للدالة السابقة، إلا أنها تعكس اتجاه دوران المحرك. تتم هذه العملية عن طريق جعل قيمة IN2 للمرحلHIGH ، و IN1 قيمة LOW

```
void motor_reverse(void)      // the function that will cause the motor to
rotate ccw
```

```
{
  digitalWrite(MOTOR_IN1, LOW);
  digitalWrite(MOTOR_IN2, HIGH);
}
```

دالةmotor_stop() ، تقوم بإيقاف المحرك تماما عن الحركة، عن طريق جعل قيمة كلا الطرفين LOW فلا يصل التيار للمحرك فيتوقف.

```
void motor_stop(void)          // the function that will cause the motor to stop rotating
{
  digitalWrite(MOTOR_IN1, LOW);
  digitalWrite(MOTOR_IN2, LOW);
}
```




تستطيع المتحكمات الدقيقة (microcontrollers) استخلاص الإشارات الرقمية الثنائية (وهما احدى قيمتين صفر أو واحد). كمثال هل الزّر مضغوط أم لا؟ هذا هو ما تعنيه الإشارات الرقمية.

إذا تم تغذية المتحكم الدقيق بجهد قيمته 5 فولت فإنه يفسر صفر فولت كقيمة ثنائية (صفر)، وخمسة فولت كقيمة ثنائية (واحد). ولكن ليس الأمر بهذه البساطة، وعادة ما يكون هناك تفاوت وتدرج في القيم. ماذا تعنى قيمة الإشارة إذا كانت 2.72 فولت؟ هل يعني ذلك أنها (صفر أم واحد)؟ غالباً ما تدفعنا الحاجة إلى قياس الإشارات المتغيرة، والتي تُسمى الإشارات التناظرية أو القياسية.

ربما يُعطى جهاز استشعار تناظري (قيمته 5 فولت) خرجاً 0.01 فولت أو 4.99 فولت أو قيمة بينهما. ولحسن الحظ عادة ما تُزود المتحكمات الدقيقة بجهاز مثبت داخلها والذي يسمح لنا بتحويل هذه الجهود إلى قيم يمكننا استخدامها فيما بعد في برنامج أخر لكي ينفذ أمراً ما.

ما هو المحول التناظري الرقمي (ADC)؟

يعتبر المحول التناظري الرقمي سمة مميزة ومفيدة؛ حيث يقوم بتحويل القيمة التناظرية للجهد على أحد الوصلات إلى قيمة رقمية. وبتحويل القيم من العالم القياسي (التناظري) إلى العالم الرقمي؛ يمكننا البدء في استخدام الإلكترونيات في التواصل مع العالم الخارجي المحيط بنا.

ليس لكل منفذ على المتحكم الدقيق القدرة على التحويل من قيم تناظرية إلى قيم رقمية. فمثلاً في بطاقة أردوينو يُرمز لهذه المنافذ بالرمز (A) بالإضافة إلى رقم (من A0 إلى A5) وذلك للإشارة إلى أن هذه المنافذ تستطيع قراءة قيم الجهد التناظرية.

وبالنظر إلى المحول التناظري الرقمي نجد أنه يختلف بشكل كبير من متحكم الدقيق إلى آخر. فالمحول الرقمي التناظري على بطاقة أردوينو له سعة 10 خانات (10(bit-، بمعنى أن لديه القدرة على استخلاص 1024 ($2^{10}$) قيمة تناظرية منفصلة. البعض الآخر من المتحكمات الدقيقة تحتوي على محولات رقمية ذات ثمان خانات ($2^8$=256 قيمة تناظرية منفصلة)، والبعض لديه 16 خانة ($2^{16}$=65535 قيمة تناظرية منفصلة).

تعتبر الطريقة التي يعمل بها المحول التناظري الرقمي معقدة بعض الشيء، هناك طرق مختلفة ولكنها قليلة لتحقيق ذلك العمل المبهر (يمكنك قراءة تلك القائمة)، ولكن الطريقة الأكثر شيوعاً تستخدم قيم تناظرية للجهد لشحن مكثف داخلي ثم قياس الوقت الذي تستغرقه عملية تفريغ المكثف خلال المقاومات داخلية. يعمل المتحكم الدقيق على مراقبة عدد الدورات لعداد التوقيت والتي تتم إلى أن يتم تفريغ المكثف. عدد الدورات هو العدد الذي تتم إعادته بعد اكتمال عملية التحويل.

يُعطي المحول الرقمي قيمة نسبية، بمعنى أنه يفترض أن قيمة خمسة فولت تقابلها 1023، والقيم التي أقل من خمسة فولت، تأخذ قيماً نسبية فيما بين خمسة فولت و1023.

$$\frac{Resolution\ of\ the\ ADC}{System\ Voltage} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured}$$

يعتمد التحويل من تناظري إلى رقمي على قيمة جهد النظام، وحيث أننا نستخدم محول رقمي بسعة عشر خانات الخاص ببطاقات أردوينو وجهد النظام قيمته 5 فولت؛ يمكننا تبسيط المعادلة السابقة كالتالي:

$$\frac{1023}{5} = \frac{ADC\ Reading}{Analog\,Voltage\ Measured}$$

إذا كان نظامك يعمل بجهد 3.3 فولت، يمكنك ببساطة تغيير 5 فولت بالقيمة الجديدة للنظام والتي هي 3.3 فولت في المعادلة السابقة. إذا كان النظام 3.3 فولت والمحول التناظري الرقمي يُعطي قراءة 512، ماذا سيكون الجهد التناظري المُقاس؟ سيكون تقريبا 1.65 فولت.

أما إذا كان الفولت التناظري 2.12 فولت، ماذا ستكون قراءة المحول التناظري الرقمي؟

$$\frac{1023}{5.00V} = \frac{x}{2.12V}$$

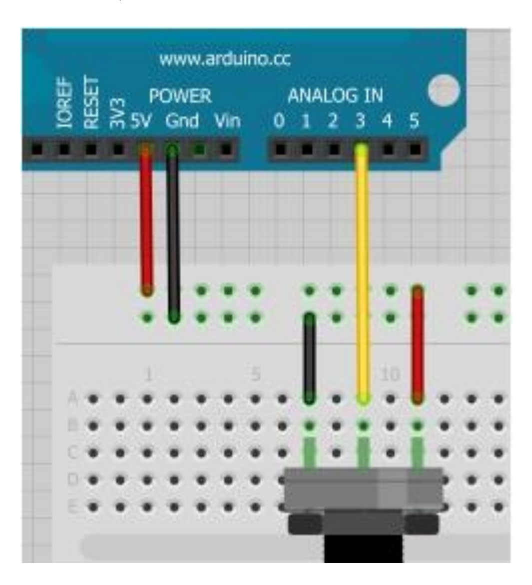
وبترتيب الحدود نجد أن:

$$\frac{1023}{5.00V} * 2.12V = x$$
$$x = 434$$

اي ان المحول الرقمي سوف يعطي قيمة 434.

مثال على التحويل التناظري الرقمي باستخدام أردوينو

لتوضيح ذلك في العالم الحقيقي، سوف نستخدم أردوينو لقياس جهد تناظري. يمكننا استخدام مقاومة متغيرة، أو جهاز استشعار للضوء أو مُجزئ للجهد؛ وذلك لإنشاء مصدر للجهد. دعونا الآن نُنشئ دائرة بسيطة باستخدام مقاومة متغيرة.




قبل البدء، لابد من تعريف المنفذ كدخل، ولكي يتماشى ذلك مع الدائرة الموضحة سوف نستخدم (A3):

```
pinMode(A3, INPUT);
```

ثم نقوم بالتحويل من تناظري إلى رقمي باستخدام الأمر analogRead() :

لقراءة القيمة التناظرية على المنفذ (A3) ويضعها في القيمة (x)

```
int x = analogRead(A3);
```

تعود القيمة ويتم تخزينها في المعامل (x) وتكون قيمته ما بين 0 إلى 1023. المحول الرقمي الموجود ببطاقة أردوينو له سعة 10 أرقام ثنائية (بمعنى أن له $2^{10} = 1024$ قيمة تناظرية مقابلة). يتم تخزين تلك القيمة في النوع (int) لأن قيمة (x) أكبر من القيمة التي تستطيع الوحدة التخزينية (byte=8 bit) الاحتفاظ بها.

دعنا نضع تلك القيمة ونرى كيف تتغير:

```
Serial.print("Analog value: ");

Serial.println(x);
```

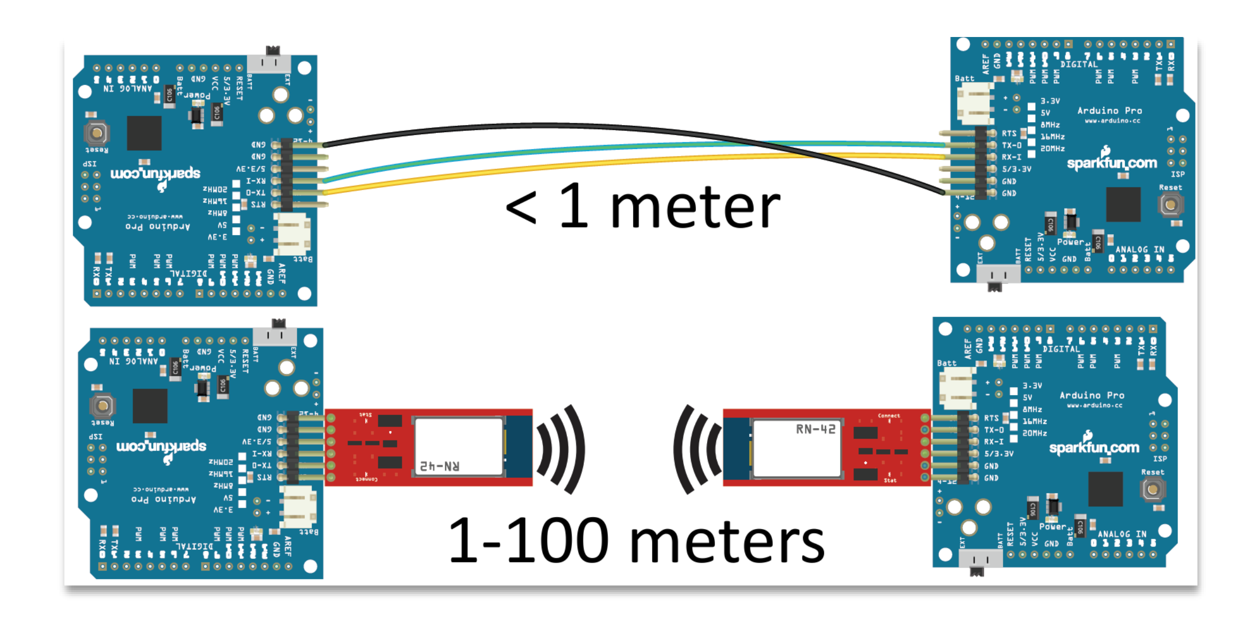
وبتغير القيمة التناظرية، القيمة (x) سوف تتغير أيضا.

فعلى سبيل المثال، إذا كانت قيمة (x) هي 334، وسوف نستخدم أردوينو بجهد 5 فولت، ماذا ستكون قيمة الفولت الحقيقي؟

وبالقياس باستخدام مقياس متعدد (مالتيميتر)، والتأكد من قيمة الفولت، من المؤكد أنها ستكون 1.63 تقريبًا.

لقد صنعت للتو جهاز رقمي متعدد القياسات باستخدام "أردوينو".

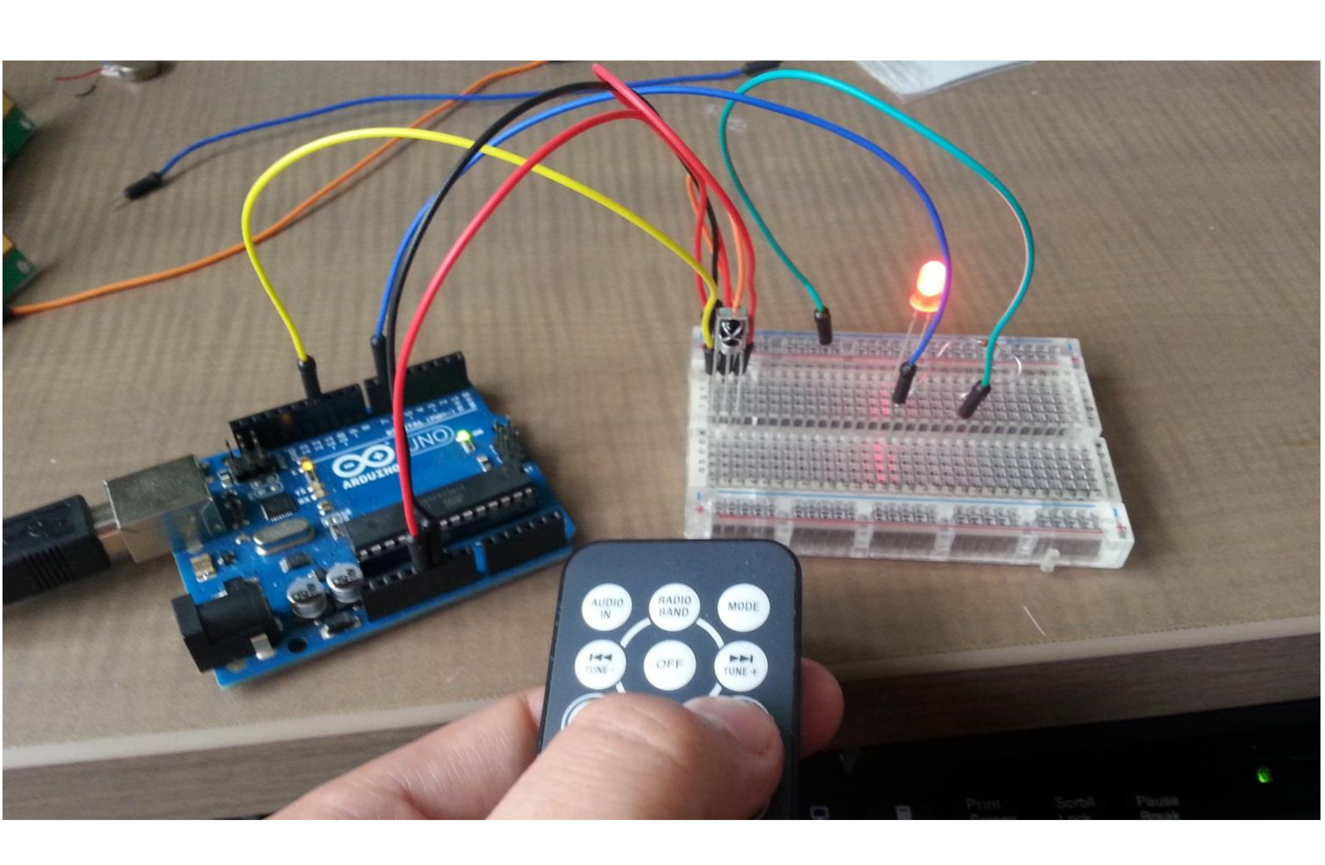
ماذا سوف يحدث إذا تم توصيل جهاز استشعار تناظري بمنفذ رقمي؟ لا شيء سيئ سوف يحدث، كل ما في الأمر أنك لن تستطيع تنفيذ الأمر analogRead.




لقد تعلمنا في الدروس السابقة كيفية استقبال وعرض المعلومات من الحساسات بواسطة الشاشة التسلسلية وارسال المعلومات ومن والى المحركات والحساسات بواسطة كيبل ال USB لكن ماذا لو اردنا ارسال هذه المعلومات لاسلكيا دون اسلاك ؟!

يدعم الاردوينو العديد من انظمة التحكم عن بعد باستخدام قطع مختلفة لجعل المشاريع اكر تفاعلية وجمالا فيمكن التحكم في اضاء مصباح صغير باستخدام برنامج على جهاز الحاسوب او الهاتف دون الحاجة الى اتصال سلكي لا بل ويمكن التحكم في منزل كاملا مثل جهاز التكيف وانظمة الانارة والمراقبة , وايضا يمكن استقبال المعلومات وعرضها على جهاز الهاتف بواسطة البلوتوث وصناعة المشاريع المتقدمة مثل التحكم في الطائرات واستخدام انظمة الاقمار الاصطناعية في تحديد المواقع مثل نظام التموضع العالمي.

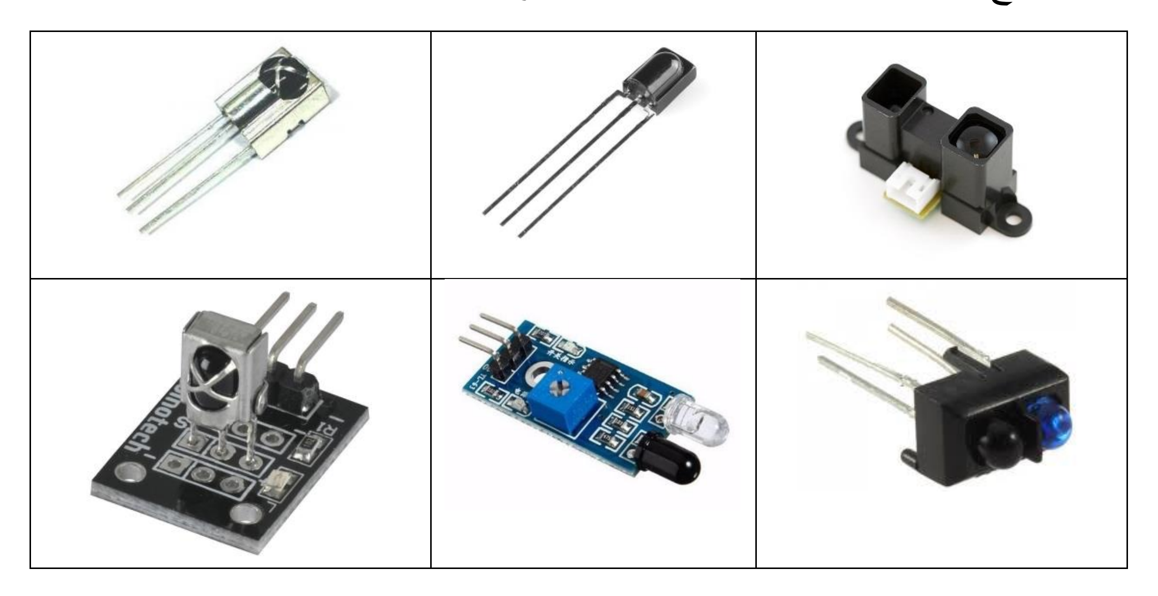
▪ اولا : التحكم عن بعد باستخدام الأشعة تحت الحمراء Infrared ويرمز لها (IR)




يعتبر الاتصال بالأشعة تحت الحمراء (IR) تقنية لاسلكية واسعة الاستخدام وسهلة التطبيق تحتوي على العديد من التطبيقات المفيدة. أبرز الأمثلة في الحياة اليومية هي أجهزة التحكم عن بعد للتلفزيون / الفيديو وأجهزة استشعار الحركة ومقاييس الحرارة بالأشعة تحت الحمراء.

هناك الكثير من مشاريع اردوينو المثيرة للاهتمام والتي تستخدم أيضًا اتصالات الأشعة تحت الحمراء. مع جهاز إرسال واستقبال الأشعة تحت الحمراء بسيط ، يمكنك جعل الروبوتات التي يتم التحكم فيها عن بعد ، وأجهزة الاستشعار عن بعد ، ومراقبة معدل ضربات القلب ، وأجهزة التحكم عن بعد الخاصة بالتلفزيون ، وغير ذلك الكثير.

بعض انواع واشكال حساس الاشعة تحت الحمراء



في هذا الدرس ، سأشرح طريقة عمل الأشعة تحت الحمراء وكيفية عملها. ثم سأوضح لك كيفية إعداد جهاز استقبال الأشعة تحت الحمراء ووحدة تحكم عن بعد على Arduino. سأوضح لك أيضًا كيفية استخدام أي جهاز تحكم عن بُعد يعمل بالأشعة تحت الحمراء (مثل جهاز التلفزيون الخاص بك) للتحكم في الأشياء المتصلة بـ Arduino.

الآن دعونا ندخل في التفاصيل , الأشعة تحت الحمراء هي شكل من أشكال الضوء يشبه الضوء الذي نراه في كل مكان حولنا. الفرق الوحيد بين ضوء الأشعة تحت الحمراء والضوء المرئي هو التردد والطول الموجي. تقع الأشعة تحت الحمراء خارج نطاق الضوء المرئي ، لذلك لا يمكن للبشر رؤيتها لذا يجب استخدام الكاميرا كاميرا الهاتف مثلاً اذا احببت ان تشاهدها.

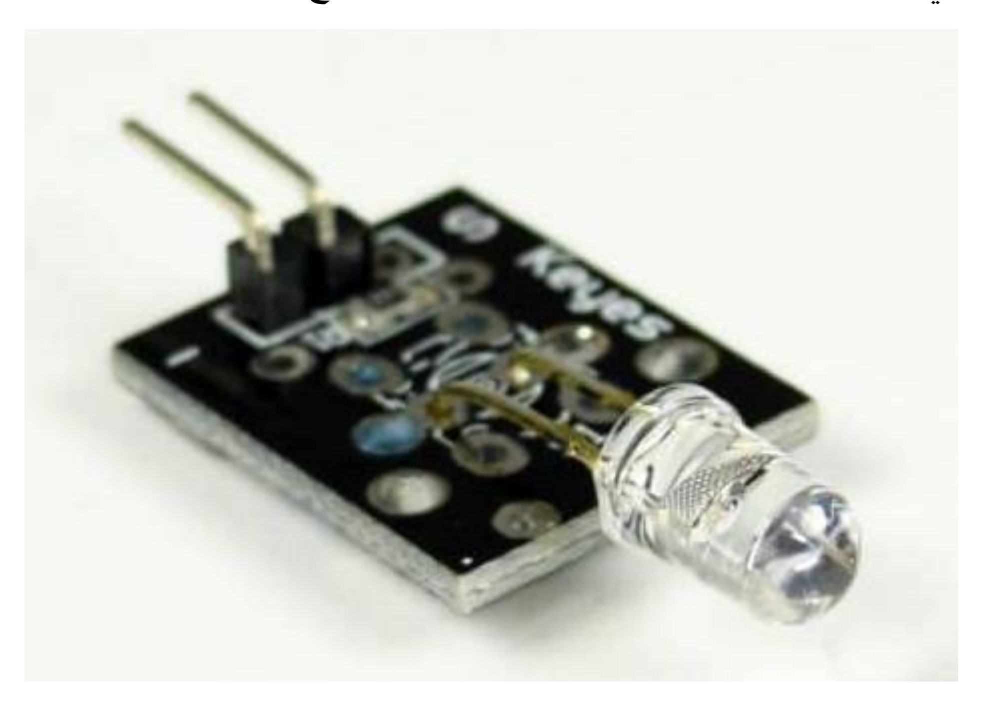
ولأن الأشعة تحت الحمراء هي نوع من الضوء ، يتطلب اتصال الأشعة تحت الحمراء خط رؤية مباشر من جهاز الاستقبال إلى جهاز الإرسال. لا يمكن نقلها عبر الجدران أو المواد الأخرى مثل WiFi أو البلوتوث.

يبدو جهاز الإرسال تمامًا مثل مصباح LED ، إلا أنه ينتج ضوءًا في طيف الأشعة تحت الحمراء بدلاً من الطيف المرئي.

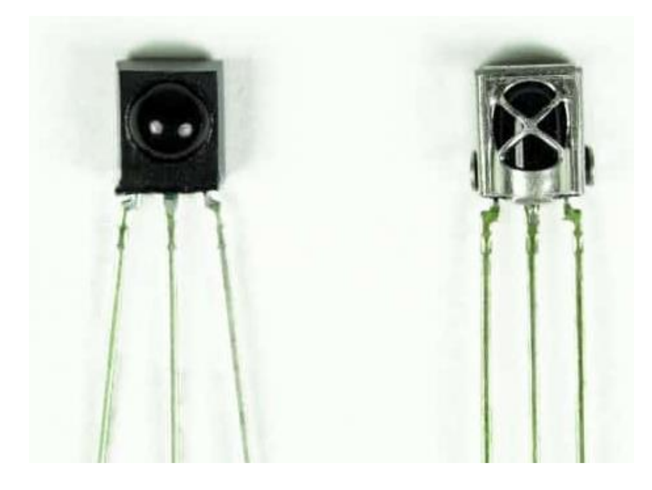
والصورة التالية تمثل الحساس الموجود في وحدة الارسال تستخدم لارسال اشارات تحت الحمراء :



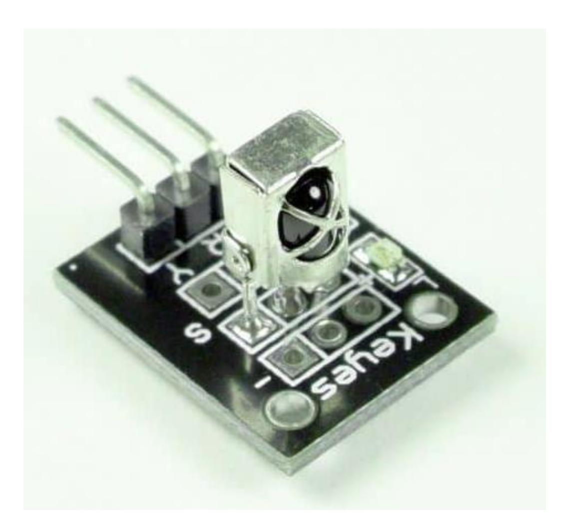

يتم استخدام نفس النوع من LED في لوحات استقبال مرسل IR للاردوينو و يمكنك رؤيتها في مقدمة جهازالاستقبال الخاص بـ لوحة المفاتيح اعلاه:



جهاز استقبال الأشعة تحت الحمراء هو ثنائي ضوئي ومكبر صوت مسبق يحول ضوء الأشعة تحت الحمراء إلى إشارة كهربائية. الثنائيات استقبال IR تبدو عادة مثل هذا:

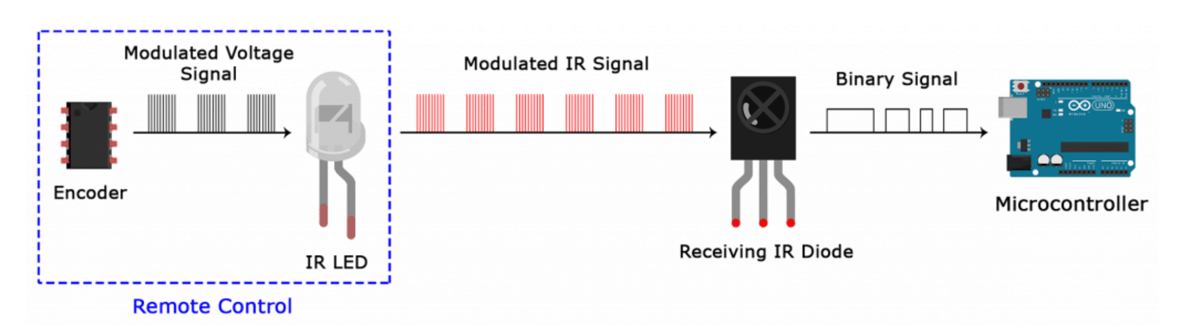


وبعض الانواع لها لوحة خاصة مثل هذا :




ينبعث ضوء الأشعة تحت الحمراء من الشمس والمصابيح الكهربائية وأي شيء آخر ينتج الحرارة. هذا يعني أن هناك الكثير من ضوضاء الأشعة تحت الحمراء في كل مكان حولنا. لمنع هذه الضوضاء من التداخل مع إشارة الأشعة تحت الحمراء ، يتم استخدام تقنية تشكيل الإشارة (signal modulation technique).

في تشكيل إشارات الأشعة تحت الحمراء ، يقوم المشفر على جهاز التحكم عن بعد بالأشعة تحت الحمراء بتحويل إشارة ثنائية إلى إشارة كهربائية معدلة. يتم إرسال هذه الإشارة الكهربائية إلى LED الإرسال. يحول مؤشر LED للتحويل الإشارة الكهربائية المعدلة إلى إشارة ضوئية تحت الحمراء معدلة. ثم يقوم جهاز استقبال الأشعة تحت الحمراء بإزالة تشكيل إشارة الأشعة تحت الحمراء وتحويلها إلى ثنائي قبل تمرير المعلومات إلى وحدة التحكم الدقيقة انظر الشكل التالي :




إشارة الأشعة تحت الحمراء المعدلة هي عبارة عن سلسلة من نبضات ضوئية تعمل بالأشعة تحت الحمراء يتم تشغيلها وإيقاف تشغيلها عند تردد عال يعرف بتكرار الموجة الحاملة. إن التردد الحامل الذي تستخدمه معظم المرسلات هو 38 kHz ، لأنه نادر في الطبيعة وبالتالي يمكن تمييزه عن الضوضاء المحيطة. وبهذه الطريقة ، سيعرف جهاز استقبال الأشعة تحت الحمراء أن إشارة 38 كيلوهرتز قد تم إرسالها من المرسل ولا يتم التقاطها من البيئة المحيطة .

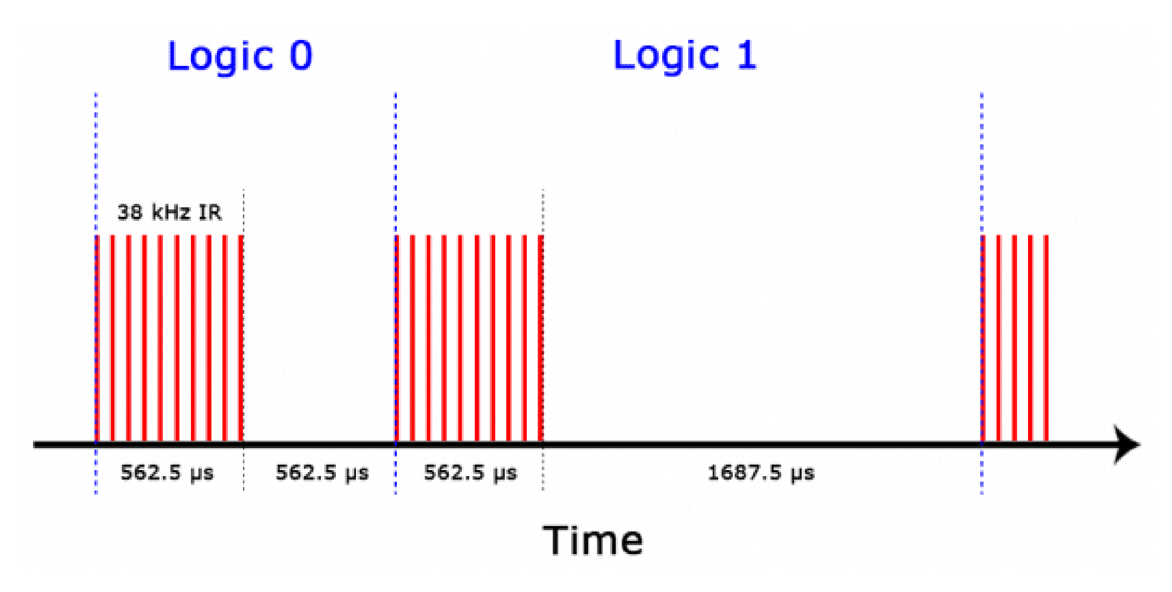
يقوم الصمام الثنائي للمستقبل بالكشف عن جميع ترددات ضوء الأشعة تحت الحمراء ، ولكنه يحتوي على مرشح تمرير النطاق ويسمح فقط من خلال الأشعة تحت الحمراء عند 38 كيلوهرتز. ثم يقوم بتضخيم الإشارة المعدلة مع مضخم مسبق وتحويلها إلى إشارة ثنائية قبل إرسالها إلى وحدة تحكم دقيقة.

بروتوكولات نقل الأشعة تحت الحمراء:

يتم تعريف النمط الذي يتم فيه تحويل إشارة الأشعة تحت الحمراء المعدلة إلى ثنائي بواسطة بروتوكول الإرسال. هناك العديد من بروتوكولات الإرسال IR. تعد Sony و Matsushita و NEC و RC5 بعض البروتوكولات الأكثر شيوعًا.

بروتوكول NEC هو أيضا النوع الأكثر شيوعا في مشاريع اردوينو ، لذلك سأستخدم كمثال ليوضح لك كيف يحول جهاز الاستقبال إشارة الأشعة تحت الحمراء المعدلة إلى ثنائي واحد (binary one) .

يبدأ المنطق "1" بنبضة عالية يبلغ طولها 562.5 µs تبلغ 38 kHz IR متبوعة بنبض LOW طويل يبلغ 1،687.5 µs. تُنقل المنطقي "0" بنبضة عالية يبلغ طولها 562.5 followeds تليها نبضة LOW طويلة 562.5 µs .




هكذا يقوم بروتوكول NEC بتشفير وفك البيانات الثنائية إلى إشارة معدلة!. تختلف البروتوكولات الأخرى فقط في مدة نبضات HIGH و LOW الفردية.

برمجة حساس استقبال الاشعة تحت :

في كل مرة تضغط فيها على زر في جهاز التحكم عن بعد ، يتم إنشاء رمز سداسي. هذه هي المعلومات التي يتم تعديلها وإرسالها عبر IR إلى جهاز الاستقبال. من أجل فك أي مفتاح تم الضغط عليه والتعرف عليه ، يحتاج المتحكم المستلم إلى معرفة الرمز الذي يتوافق مع كل مفتاح على جهاز التحكم عن بعد.

ترسل أجهزة التحكم عن بُعد رموزًا مختلفة عند الضغط على المفاتيح ، لذا ستحتاج إلى تحديد الشفرة التي تم إنشاؤها لكل مفتاح على جهاز التحكم عن بُعد الخاص بك. إذا تمكنت من العثور على ورقة البيانات ، فيجب إدراج رموز مفاتيح الأشعة تحت

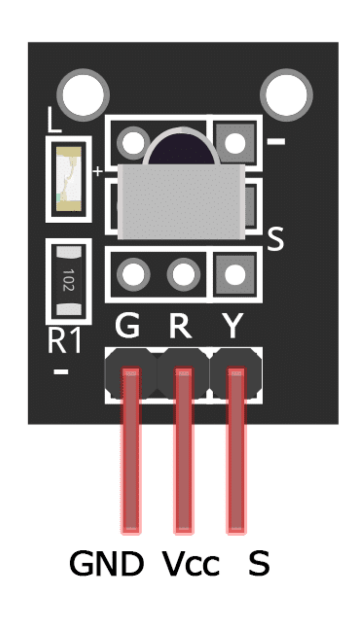
الحمراء. إذا لم يكن كذلك ، هناك رسم تخطيطي بسيط لاردوينو والذي سيقرأ معظم أجهزة التحكم عن بعد المعروفة ويطبع الشفرات السداسية العشرية إلى الشاشة التسلسلية عند الضغط على أحد المفاتيح. سأوضح لك كيفية إعداد ذلك ، ولكن نحتاج أولاً إلى توصيل جهاز الاستقبال بـ Arduino

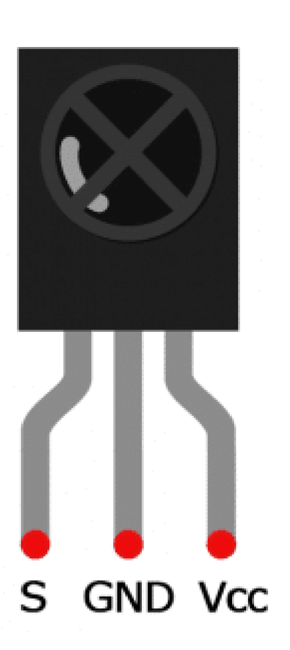
كيفية توصيل جهاز استقبال IR إلى ARDUINO لمعرفة الشيفرة لكل زر :

هناك عدة أنواع مختلفة من أجهزة استقبال الأشعة تحت الحمراء ، بعضها قائم بذاته ، وبعضها مثبت على لوحة جانبية كما بينا ذلك سابقا. تحقق من ورقة البيانات لمستقبلك IR الخاص حيث قد يتم ترتيب المسامير بشكل مختلف عن جهاز استقبال الأشعة تحت الحمراء .

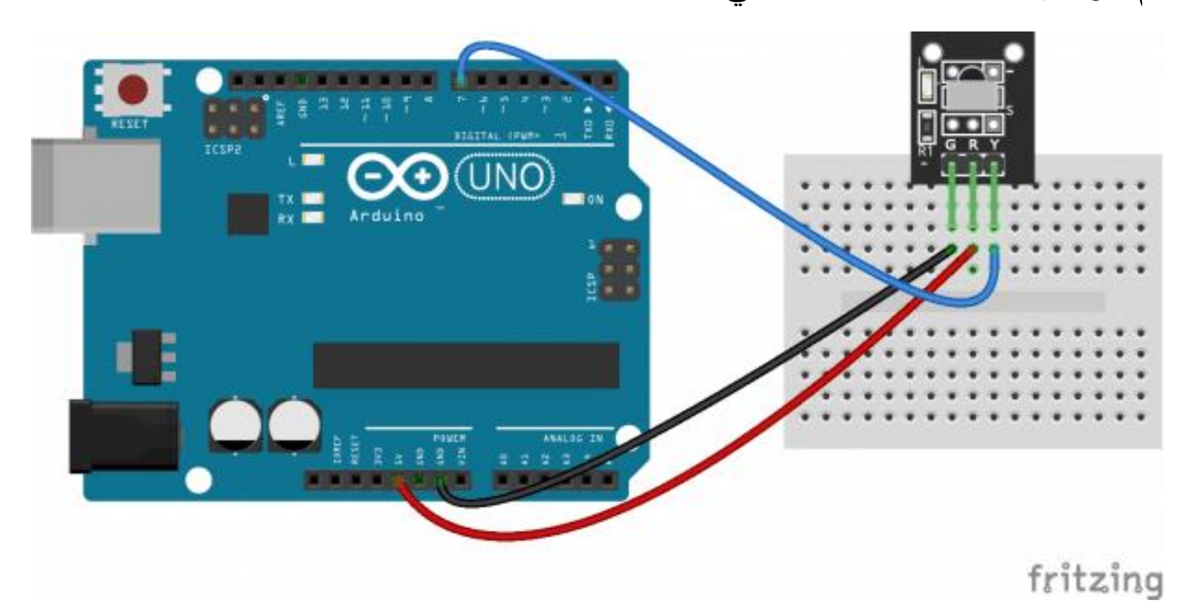
وجهاز التحكم عن بعد الذي أستخدمه هنا هو (HX1838). ومع ذلك ، سيكون لكل مستقبلات الأشعة تحت الحمراء ثلاثة دبابيس: إشارة ، أرضي ، و Vcc.

التعرف على اطراف الحساس المستقبل :

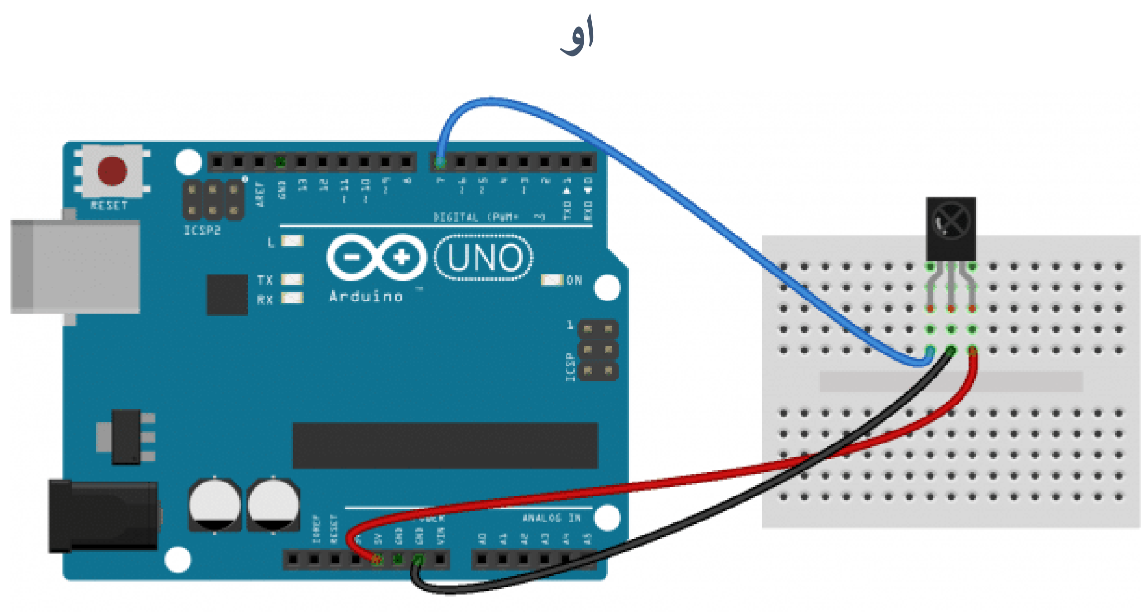


قم بتوصيله حسب المخطط التالي :




او




بمجرد توصيل جهاز الاستقبال ، يمكننا تثبيت مكتبة Arduino وبدء البرمجة. في الأمثلة أدناه ، سأوضح كيفية العثور على الرموز المرسلة بواسطة جهاز التحكم عن بُعد ، وكيفية العثور على بروتوكول IR الذي يستخدمه جهاز التحكم عن بُعد ، وكيفية طباعة المطابع على الشاشة التسلسلية ، وأخيرًا ، كيفية التحكم دبابيس الإخراج اردوينو مع جهاز تحكم عن بعد.

سنستخدم مكتبة IRremote لجميع أمثلة الكودات أدناه. يمكنك تنزيل ملف ZIP للمكتبة من هنا :

/http://z3t0.github.io/Arduino-IRremote

او من محتويات قرص الكتاب .

لتثبيت المكتبة من ملف ZIP ، افتح Arduino IDE ، ثم انتقل إلى Sketch> Include Library> Add .ZIP Library ، ثم حدد ملف IRremote ZIP الذي قمت بتنزيله .

للعثور على رموز المفاتيح لجهاز التحكم عن بعد ، قم بتحميل هذا الكود إلى Arduino وافتح الشاشة التسلسلية:

```
#include <IRremote.h>

const int RECV_PIN = 7;

IRrecv irrecv(RECV_PIN);

decode_results results;

void setup(){

  Serial.begin(9600);

  irrecv.enableIRIn();

  irrecv.blink13(true);

}

void loop(){

  if (irrecv.decode(&results)){

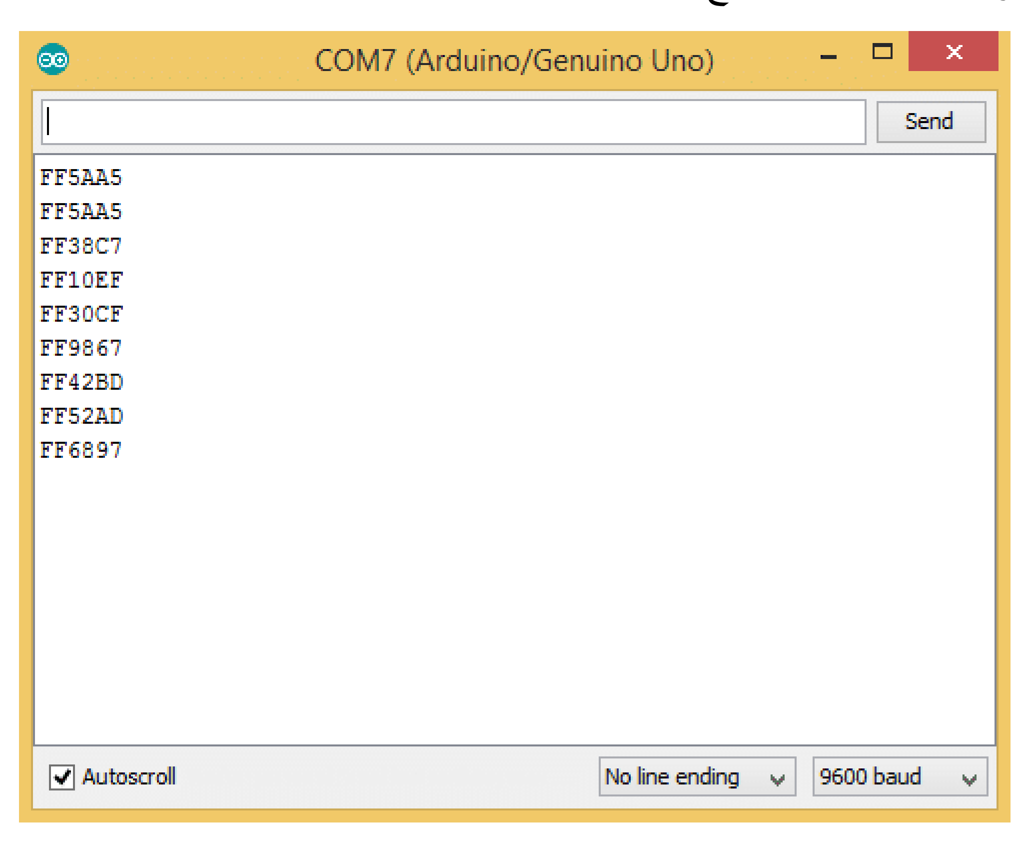
        Serial.println(results.value, HEX);

        irrecv.resume();

  }

}
```

الآن اضغط على كل مفتاح على جهاز التحكم عن بعد وقم بتسجيل الشفرة السداسية المطبوعة لكل ضغطة مفتاح.

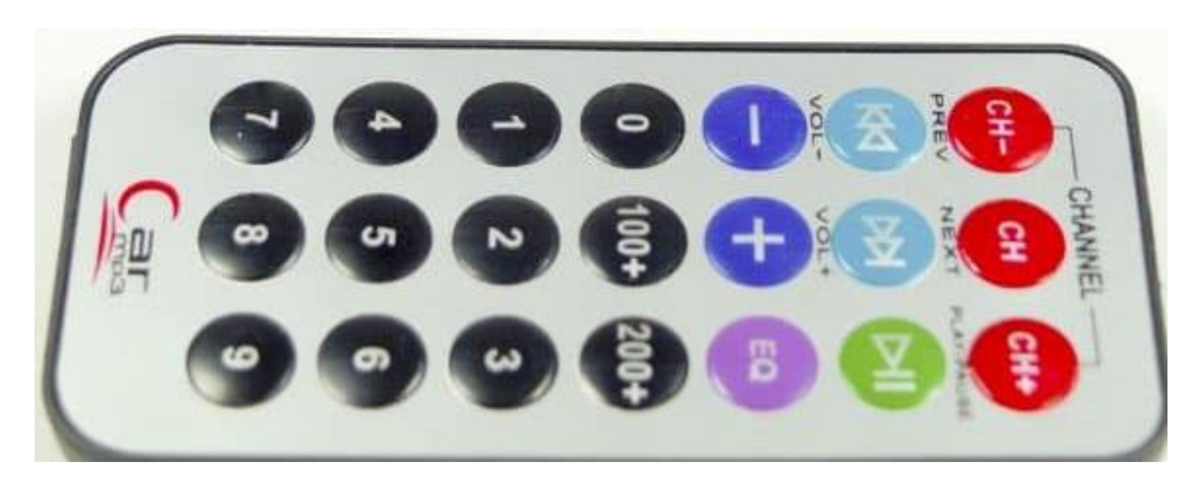



باستخدام البرنامج أعلاه ، حصلنا على الجدول التالي الذي يبين المفاتيح ورموزها المقابلة من جهاز التحكم عن بعد الذي جاء مع جهاز استقبال الأشعة تحت الحمراء HX1838. لاحظ أن تتلقى رمز XFFFFFFFF0 عند ضغط مفتاح باستمرار.

| Key | Code |
|---|---|
| CH- | 0xFFA25D |
| CH | 0xFF629D |
| CH+ | 0xFFE21D |
| << | 0xFF22DD |

<table>
<tr><td>>></td><td>0xFF02FD</td></tr>
<tr><td>>||</td><td>0xFFC23D</td></tr>
<tr><td>–</td><td>0xFFE01F</td></tr>
<tr><td>+</td><td>0xFFA857</td></tr>
<tr><td>EQ</td><td>0xFF906F</td></tr>
<tr><td>100+</td><td>0xFF9867</td></tr>
<tr><td>200+</td><td>0xFFB04F</td></tr>
<tr><td>0</td><td>0XFF6897</td></tr>
<tr><td>1</td><td>0xFF30CF</td></tr>
<tr><td>2</td><td>0xFF18E7</td></tr>
<tr><td>3</td><td>0xFF7A85</td></tr>
<tr><td>4</td><td>0xFF10EF</td></tr>
<tr><td>5</td><td>0xFF38C7</td></tr>
<tr><td>6</td><td>0xFF5AA5</td></tr>
<tr><td>7</td><td>0xFF42BD</td></tr>
<tr><td>8</td><td>0xFF4AB5</td></tr>
<tr><td>9</td><td>0xFF52A</td></tr>
</table>

شكل جهاز الارسال موضحا ازراره :




كيفية معرفة نوع البروتوكول المستخدم من قبل جهاز التحكم عن بعد :

معرفة أي بروتوكول تستخدمه عن بُعد يمكن أن يكون مفيدًا إذا كنت ترغب في العمل على بعض المشاريع الأكثر تقدمًا. أو قد تكون مجرد فضول. سيحدد البرنامج أدناه البروتوكول الذي يستخدمه جهاز التحكم عن بعد الخاص بك. يعمل حتى على معظم أجهزة التحكم عن بعد في منزلك.

لمعرفة ذلك نستخدم الكود التالي :

```
#include <IRremote.h>



const int RECV_PIN = 7;

IRrecv irrecv(RECV_PIN);

decode_results results;



void setup(){

  Serial.begin(9600);

  irrecv.enableIRIn();

  irrecv.blink13(true);
```

```
}
void loop(){
  if (irrecv.decode(&results)){
        Serial.println(results.value, HEX);
        switch (results.decode_type){
            case NEC: Serial.println("NEC"); break ;
            case SONY: Serial.println("SONY"); break ;
            case RC5: Serial.println("RC5"); break ;
            case RC6: Serial.println("RC6"); break ;
            case DISH: Serial.println("DISH"); break ;
            case SHARP: Serial.println("SHARP"); break ;
            case JVC: Serial.println("JVC"); break ;
            case SANYO: Serial.println("SANYO"); break ;
            case MITSUBISHI: Serial.println("MITSUBISHI"); break ;
            case SAMSUNG: Serial.println("SAMSUNG"); break ;
            case LG: Serial.println("LG"); break ;
            case WHYNTER: Serial.println("WHYNTER"); break ;
            case AIWA_RC_T501: Serial.println("AIWA_RC_T501"); break ;
            case PANASONIC: Serial.println("PANASONIC"); break ;
            case DENON: Serial.println("DENON"); break ;
          default:
            case UNKNOWN: Serial.println("UNKNOWN"); break ;
          }
        irrecv.resume();
 }
}
```

الان نريد طباعة قيمة المفتاح بدلاً من الشفرة السداسية نستخدم الكود التالي :

```
#include <IRremote.h>


const int RECV_PIN = 7;

IRrecv irrecv(RECV_PIN);

decode_results results;

unsigned long key_value = 0;


void setup(){

  Serial.begin(9600);

  irrecv.enableIRIn();

  irrecv.blink13(true);

}


void loop(){

  if (irrecv.decode(&results)){


        if (results.value == 0XFFFFFFFF)

          results.value = key_value;


        switch(results.value){

          case 0xFFA25D:

          Serial.println("CH-");

          break;

          case 0xFF629D:

          Serial.println("CH");
```

```
break;
case 0xFFE21D:
Serial.println("CH+");
break;
case 0xFF22DD:
Serial.println("|<<");
break;
case 0xFF02FD:
Serial.println(">>|");
break ;
case 0xFFC23D:
Serial.println(">|");
break ;
case 0xFFE01F:
Serial.println("-");
break ;
case 0xFFA857:
Serial.println("+");
break ;
case 0xFF906F:
Serial.println("EQ");
break ;
case 0xFF6897:
Serial.println("0");
break ;
case 0xFF9867:
```

```
Serial.println("100+");
break ;
case 0xFFB04F:
Serial.println("200+");
break ;
case 0xFF30CF:
Serial.println("1");
break ;
case 0xFF18E7:
Serial.println("2");
break ;
case 0xFF7A85:
Serial.println("3");
break ;
case 0xFF10EF:
Serial.println("4");
break ;
case 0xFF38C7:
Serial.println("5");
break ;
case 0xFF5AA5:
Serial.println("6");
break ;
case 0xFF42BD:
Serial.println("7");
break ;
```

```
          case 0xFF4AB5:

          Serial.println("8");

          break ;

          case 0xFF52AD:

          Serial.println("9");

          break ;

        }

        key_value = results.value;

        irrecv.resume();

  }

}
```

إذا كان جهاز التحكم عن بعد يرسل رموزًا مختلفة عن تلك الموجودة في الجدول أعلاه ، فاستبدل الشفرة السداسية في كل سطر حيث تقول مثلا:

```
case 0xFFA25D:

Serial.println(“CH-“);
```

كيف يعمل الكود :

بالنسبة لأي اتصال من خلال IR باستخدام مكتبة IRremote ، نحتاج أولاً إلى إنشاء كائن يسمى irrecv وتحديد رقم التعريف الشخصي الذي يتصل به مستقبل الأشعة تحت الحمراء

```
IRrecv irrecv(RECV_PIN);
```

هذا الكائن سوف يعتني ببروتوكول ومعالجة المعلومات من المتلقي.

الخطوة التالية هي إنشاء كائن يسمى results ، من فئة decode_results ، والتي سيتم استخدامها بواسطة كائن irrecv لمشاركة المعلومات المشفرة مع التطبيق الخاص بنا

```
decode_results results;
```

في void setup () ، نقوم أولاً بتكوين معدل سرعة بث العرض التسلسلي. بعد ذلك نبدأ جهاز استقبال الأشعة تحت الحمراء عن طريق استدعاء الوظيفة IRrecv enableIRIn ()

```
  Serial.begin(9600);

  irrecv.enableIRIn();
```

وظيفة :

```
  irrecv.blink13(true);
```

سوف تومض LED بلوحة Arduino على اللوحة في كل مرة يحصل فيها المتلقي على إشارة من جهاز التحكم عن بعد ، وهو أمر مفيد في عملية التصحيح.

في كتلة void loop () ، ستعرض الدالة irrecv.decode true إذا تم تلقي رمز وسيقوم البرنامج بتنفيذ التعليمة البرمجية في العبارة if. يتم تخزين الرمز المستلم في results.value. ثم استخدمت switch للتعامل مع كل رمز IR وطباعة قيمة المفتاح المقابلة.

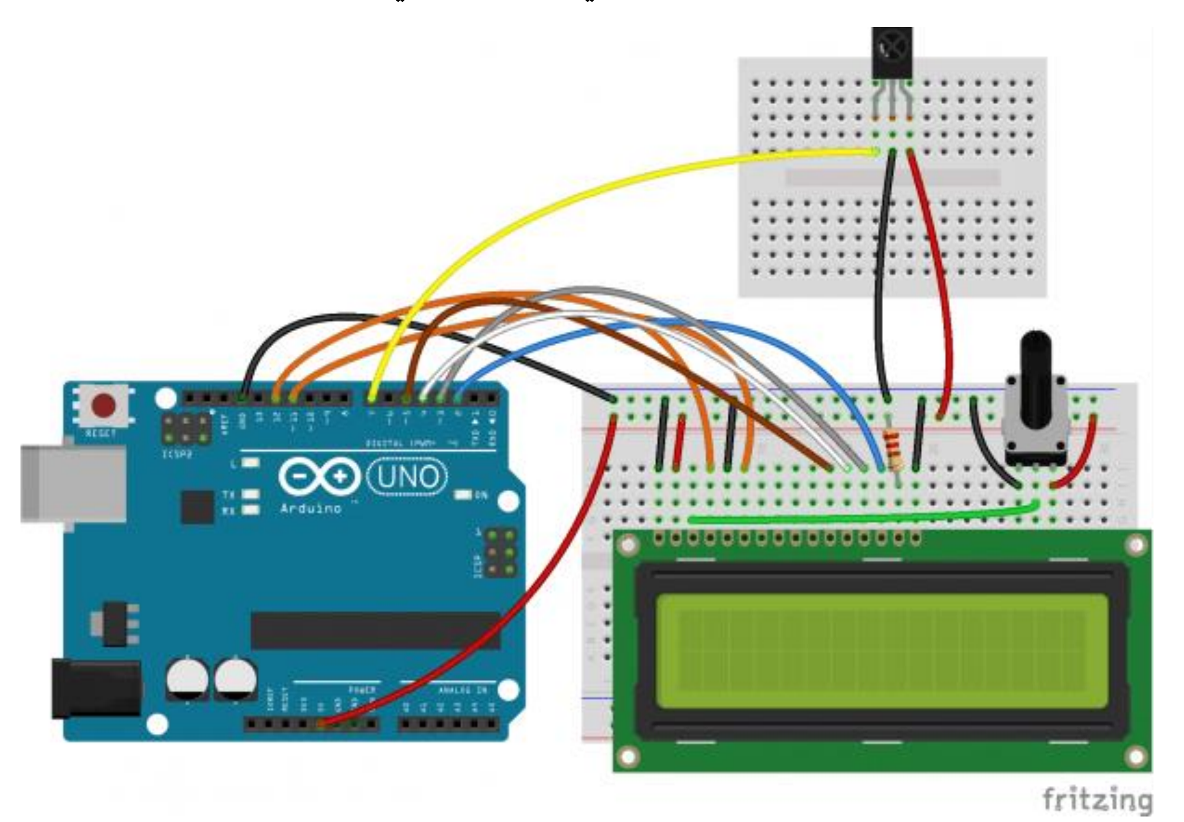
```
if (results.value == 0XFFFFFFFF)
```

```
results.value = key_value;
```

في نهاية المقطع () loop ، نسميه () irrecv.resume لإعادة ضبط جهاز الاستقبال وإعداده لتلقي الرمز التالي.

### طباعة المفاتيح إلى شاشات الكريستال السائل LCD

بدلاً من طباعة قيم المفتاح إلى جهاز العرض التسلسلي ، يمكنك أيضًا عرض المعلومات على شاشة LCD. راجع درس الشاشات حول إعداد وبرمجة شاشة LCD على LCD Arduino ، ولكن الإعداد الأساسي سيبدو كالتالي:




تتحكم المقاومة بسطوع الإضاءة الخلفية لشاشة LCD. يمكن أن يكون أي شيء من 200 أوم إلى حوالي 2 كيلو أوم. يحدد مقياس الجهد تباين الأحرف. أنا عادة استخدام K10 أوم .

بمجرد توصيل كل شيء ، قم بتحميل هذا الكود على Arduino:

```
#include <IRremote.h>

#include <LiquidCrystal.h>


const int RECV_PIN = 7;

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

IRrecv irrecv(RECV_PIN);

decode_results results;

unsigned long key_value = 0;


void setup(){

  Serial.begin(9600);

  irrecv.enableIRIn();

  irrecv.blink13(true);

  lcd.begin(16, 2);

}


void loop(){

  if (irrecv.decode(&results)){


        if (results.value == 0XFFFFFFFF)

        results.value = key_value;

        lcd.setCursor(0, 0);

        lcd.clear();


        switch(results.value){
```

```
case 0xFFA25D:

lcd.print("CH-");

break;

case 0xFF629D:

lcd.print("CH");

break;

case 0xFFE21D:

lcd.print("CH+");

break;

case 0xFF22DD:

lcd.print("|<<");

break;

case 0xFF02FD:

lcd.print(">>|");

break ;

case 0xFFC23D:

lcd.print(">|");

break ;

case 0xFFE01F:

lcd.print("-");

break ;

case 0xFFA857:

lcd.print("+");

break ;

case 0xFF906F:

lcd.print("EQ");
```

```
break ;

case 0xFF6897:

lcd.print("0");

break ;

case 0xFF9867:

lcd.print("100+");

break ;

case 0xFFB04F:

lcd.print("200+");

break ;

case 0xFF30CF:

lcd.print("1");

break ;

case 0xFF18E7:

lcd.print("2");

break ;

case 0xFF7A85:

lcd.print("3");

break ;

case 0xFF10EF:

lcd.print("4");

break ;

case 0xFF38C7:

lcd.print("5");

break ;

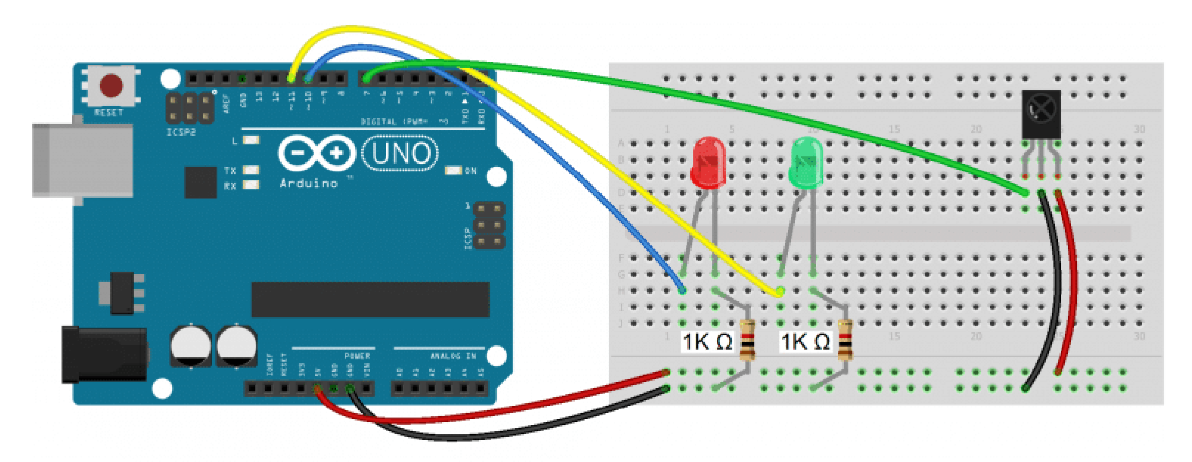
case 0xFF5AA5:
```

```
          lcd.print("6");
          break ;
          case 0xFF42BD:
          lcd.print("7");
          break ;
          case 0xFF4AB5:
          lcd.print("8");
          break ;
          case 0xFF52AD:
          lcd.print("9");
          break ;
        }
        key_value = results.value;
        irrecv.resume();
  }
}
```

واخيرا استخدام الأشعة تحت الحمراء للتحكم في الأشياء

سنعرض لك الآن عرضًا بسيطًا عن كيفية استخدام جهاز التحكم عن بُعد للأشعة تحت الحمراء للتحكم في دبابيس الإخراج في Arduino. في هذا المثال ، سنضيء مصباح LED عند الضغط على زر معين. يمكنك بسهولة تعديل الكود للقيام بأشياء مثل التحكم في محركات السيرفو ، أو تنشيط المرحلات باستخدام أي زر اضغط من جهاز التحكم عن بعد.

تحتوي هذه الدائرة على مستقبل الأشعة تحت الحمراء متصل بـ Arduino ، مع مصباح LED أحمر متصل بـ pin 10 ومصباحLED اخضر متصل بالرقم 11:




الكود أدناه للتحكم رقميًا لشتغيل المصباح HIGH لمدة ثانيتين عند الضغط على الزر "5" ، والدبوس رقمي 11 HIGH لمدة ثانيتين عند الضغط على الزر "2"

```
#include <IRremote.h>


const int RECV_PIN = 7;

IRrecv irrecv(RECV_PIN);

decode_results results;

const int redPin = 10;

const int greenPin = 11;




void setup(){

  irrecv.enableIRIn();

  irrecv.blink13(true);

  pinMode(redPin, OUTPUT);

  pinMode(greenPin, OUTPUT);

}
```

```
void loop(){
    if (irrecv.decode(&results)){


        switch(results.value){
          case 0xFF38C7: //Keypad button "5"
          digitalWrite(redPin, HIGH);
          delay(2000);
          digitalWrite(redPin, LOW);
          }

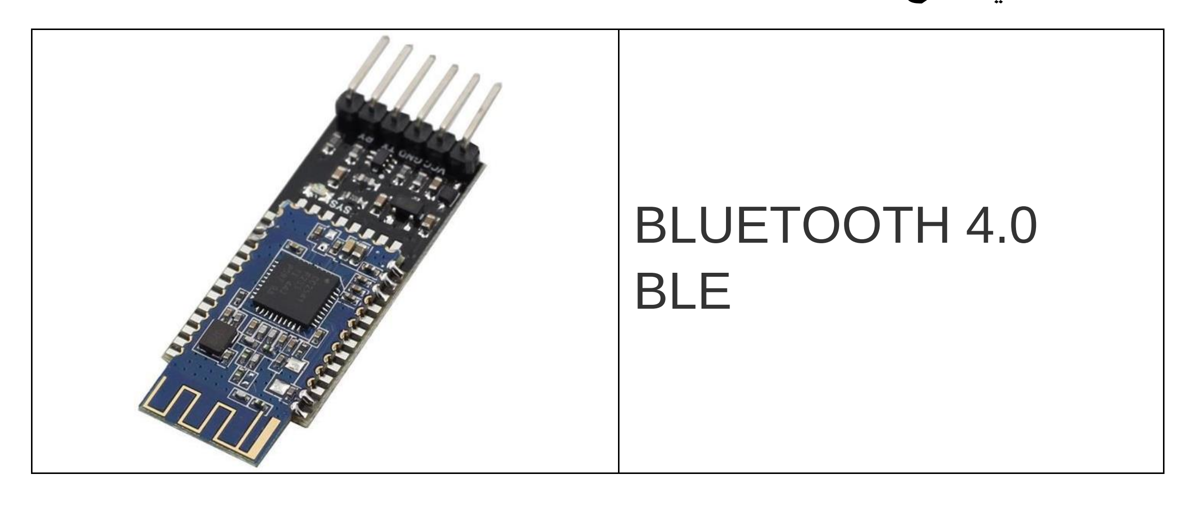

        switch(results.value){
          case 0xFF18E7: //Keypad button "2"
          digitalWrite(greenPin, HIGH);
          delay(2000);
          digitalWrite(greenPin, LOW);
          }


        irrecv.resume();
    }
}
```

لقد قمنا حتى الآن بتغطية خصائص الأشعة تحت الحمراء وكيفية حدوث الاتصال بين جهاز الإرسال وجهاز الاستقبال. لقد رأينا كيفية التعرف على رموز مفتاح IR لجهاز تحكم عن بعد محدد. تعلمنا كيفية عرض المطابع الرئيسية على الشاشة التسلسلية وعلى شاشة LCD. وأخيرًا ، أوضحت لك كيفية التحكم في خرج Arduino باستخدام جهاز التحكم عن بُعد. استمتع بالتعامل مع هذه التقنية .

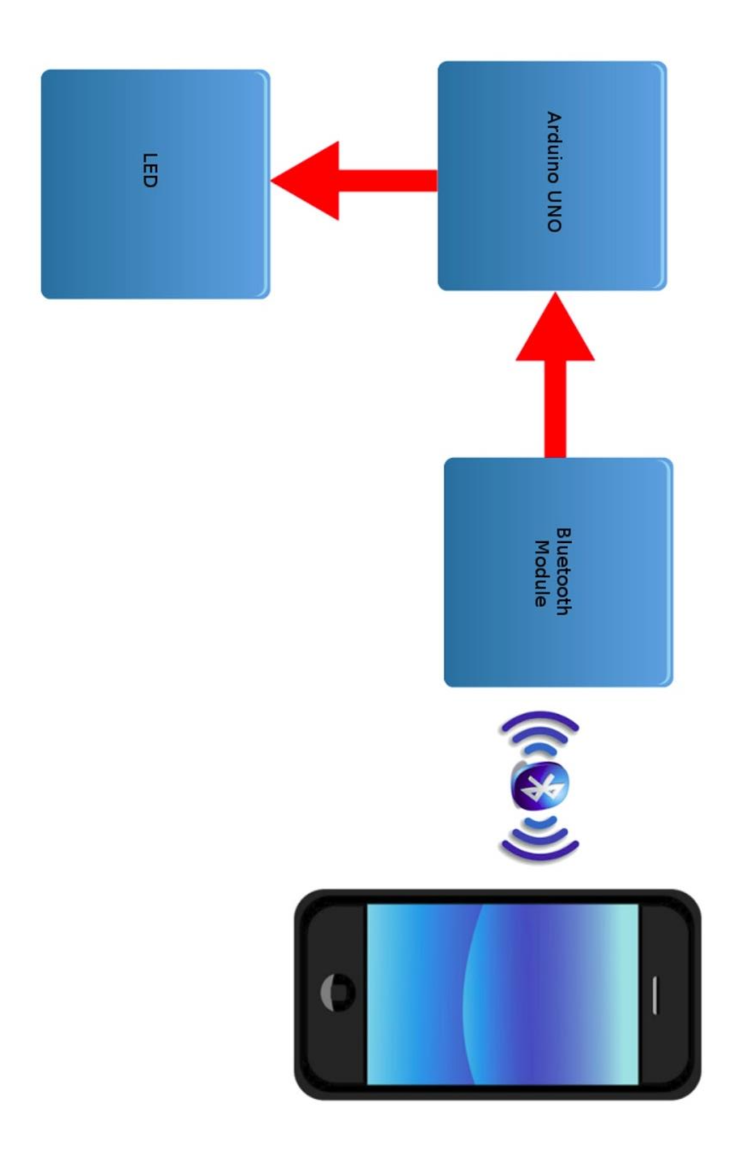
- ثانيا: التحكم عن بعد باستخدام البلوتوث Bluetooth ويرمز له (BT)

سوف نتعلم في هذا الدرس استخدام تقنية التحكم عن بعد البلوتوث للتحكم في الاردوينو باستخدام جهاز الحاسوب او الهاتف !
الجدول التالي انواع البلوتوث المختلفة

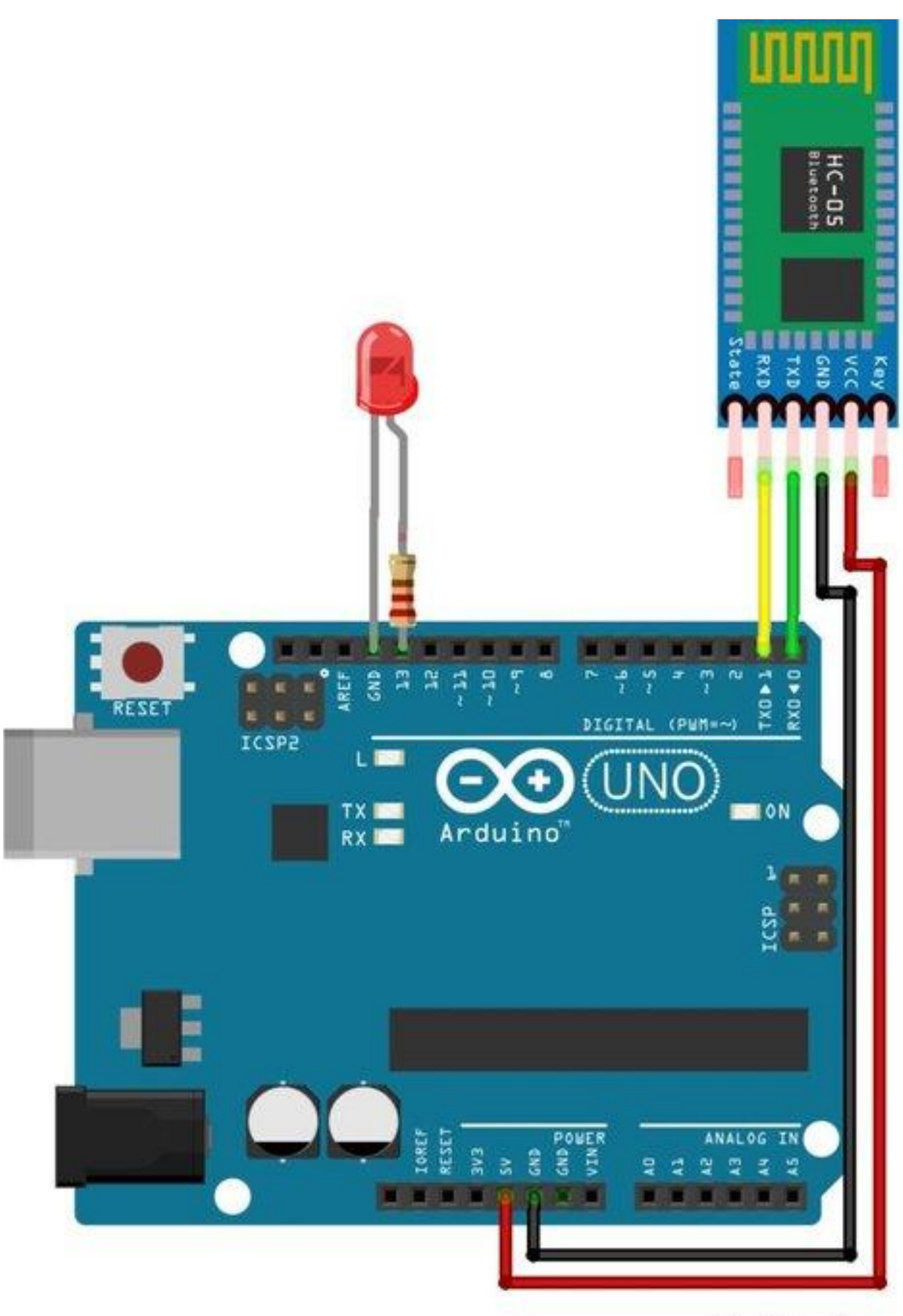
| | |
|---|---|
| BLUETOOTH 4.0 BLE |  |

| | |
|---|---|
| | BLUETOOTH MODULE<br>HC-05 |
| | HC08 BLUETOOTH V4.0 |
| | BLUETOOTH MODULE<br>HM11- CC2540 |

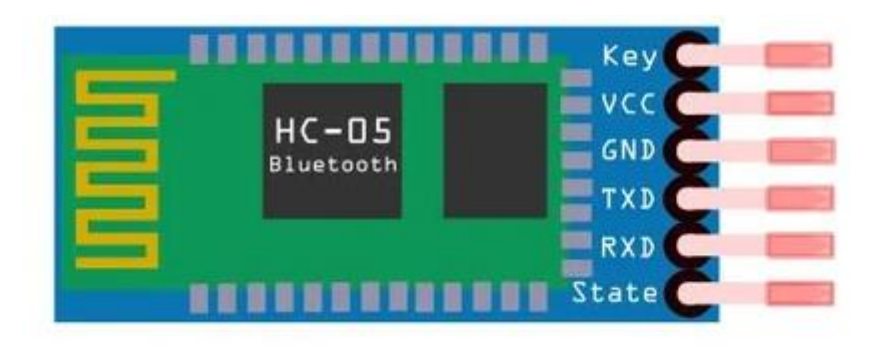
يمكن استخدام البلوتوث للتحكم في العديد من المهام مثل الانارة والروبوتات وغيرها في هذا المثال سوف نقوم بالتحكم في اضاءة مصباح LED باستخدام هاتف ذكي يعمل بنظام الاندرويد وقطعة البلوتوث من الاصدار الاكثر استخداما BLUETOOTH MODULE HC-05 وباستخدام مقاومة بقيمة 220 اوم الى 1 كيلو اوم .


LED
Arduino UNO
Bluetooth Module

قم بتوصيل الدائرة كالتالي :



كما تلاحظ الدائرة بسيطة وصغيرة ، لا يوجد سوى عدد قليل من التوصيلات وهي حسب المخطط كالتالي :




| Arduino Pins | Bluetooth Module Pins |
|---|---|
| RX (Pin 0) | TX |
| TX (Pin 1) | RX |
| 5V | VCC |
| GND | GND |

ثم نقوم بتحميل الكود التالي الى الاردوينو

```
/* Bluetooh Basic: LED ON OFF - Avishkar

* Coder - Mayoogh Girish

* Website - http://bit.do/Avishkar

* Download the App : https://github.com/Mayoogh/Arduino-Bluetooth-Basic

* This program lets you to control a LED on pin 13 of arduino using a
bluetooth module

*/

char data = 0; //Variable for storing received data

void setup()

{

    Serial.begin(9600); //Sets the baud for serial data transmission
```

```
    pinMode(13, OUTPUT); //Sets digital pin 13 as output pin

}

void loop()

{

   if(Serial.available() > 0)  // Send data only when you receive data:

   {

      data = Serial.read();        //Read the  incoming  data and store it
into variable data

      Serial.print(data);          //Print Value inside data in Serial
monitor

      Serial.print("\n");          //New line

      if(data == '1')              // Checks whether value of data is equal
to 1

         digitalWrite(13, HIGH);   //If value is 1 then LED turns ON

      else if(data == '0')         //  Checks  whether value of data is equal
to 0

         digitalWrite(13, LOW);    //If value is 0 then LED turns OFF

   }

}
```

كيف يعمل المصباح :

اصدارات البلوتوث (HC 05/06) تعمل باستخدام الاتصال التسلسلي.حيث يتم تصميم التطبيق عن طريق إرسال البيانات التسلسلية إلى وحدة بلوتوث المرسل عند الضغط على زر معين. وحدة بلوتوث في الطرف الآخر تستقبل البيانات وإدخالها إلى ardunio من خلال TX دبوس وحدة بلوتوث ثم نستخدم اردوينو للتحقق من البيانات الواردة والمقارنة إذا كانت البيانات الواردة هي 1 سوف يضيئ المصباح واذا كان 0 سوف يطفئ المصباح.

سوف نرى لك عندما نعمل اتصال بين الاردوينو والهاتف , الان سنقوم بتنزيل البرنامج المسؤول عن التحكم في جهاز الاندرويد والذي يحمل اسم LED Controller

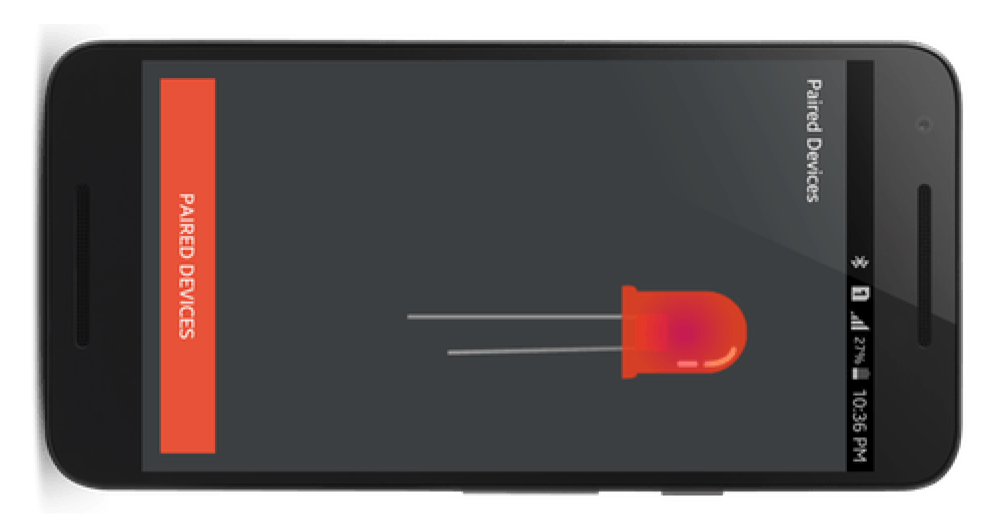
عن طريق الموقع التالي :

**www.amazon.com/Mayoogh-Girish-LED-Controller/dp/B01DR5T226/ref**

ايقونة التطبيق هي بالشكل التالي :



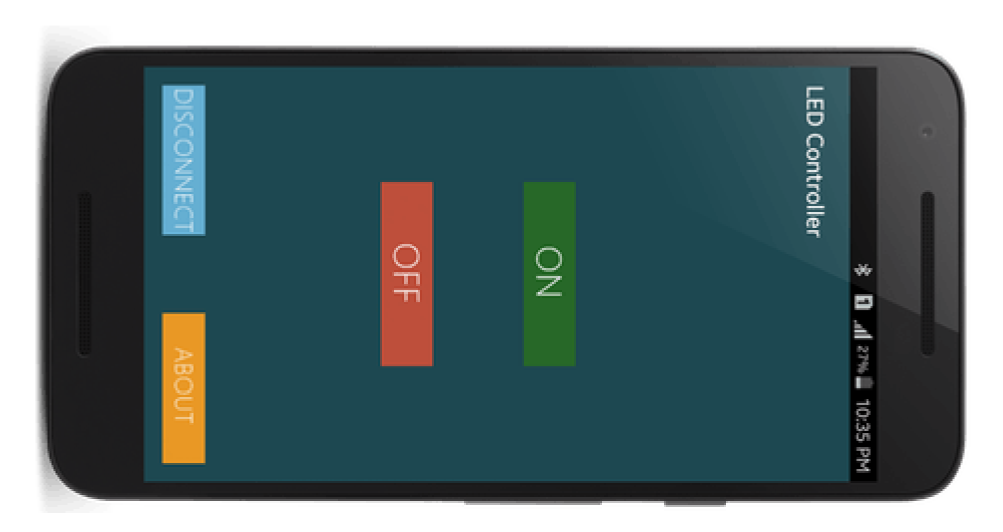
الان قم بتوصيل المخطط اعلاه , وبعد تحميل البرنامج قم بفتحه لترى ما يلي :




الان نريد عم اتصال بين الهاتف والاردوينو وذلك يتم من خلال البحث عن البلوتوث باستخدام الهاتف وسيجد الهاتف البقطعة باسم الاصدار وهي HC-05 Bluetooth

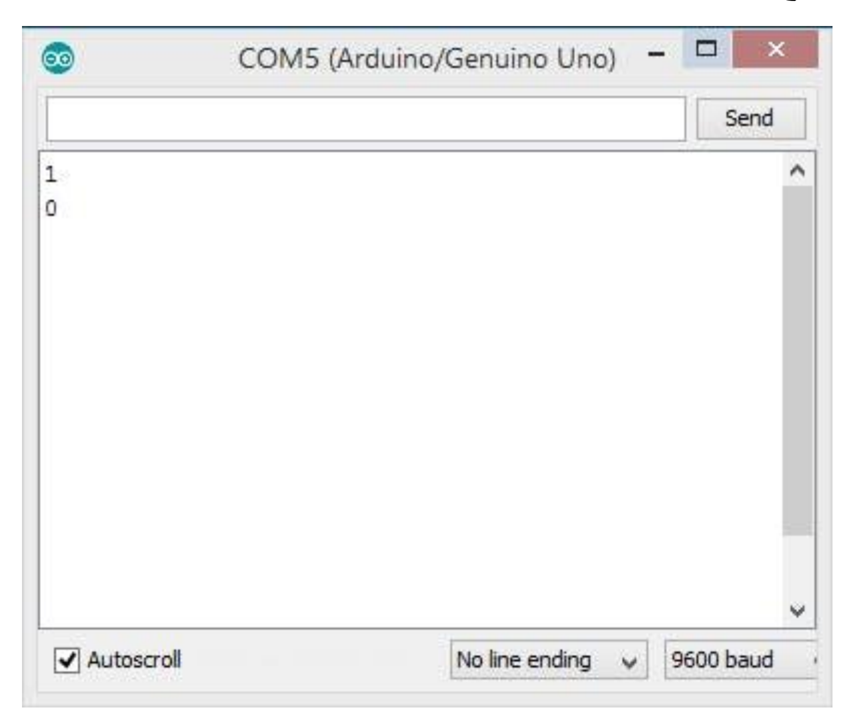
وعن عمل اقتران Pair سيطلب منك ادخال الرقم السري , ادخل 1234 او 0000

الان بعد عمل الاقتران بين الهاتف والاردوينو نذهب باستخدام التطبيق الى النافذة التالية :




قم بالضغط على ON ستلاحظ اضاءة المصباح وعند على OFF ستلاحظ اطفائه.

وعن فتح النافذة التسلسلية في بيئة الاردوينو IDE ستلاحظ ما يلي عند تشغيل واطفاء المصباح :

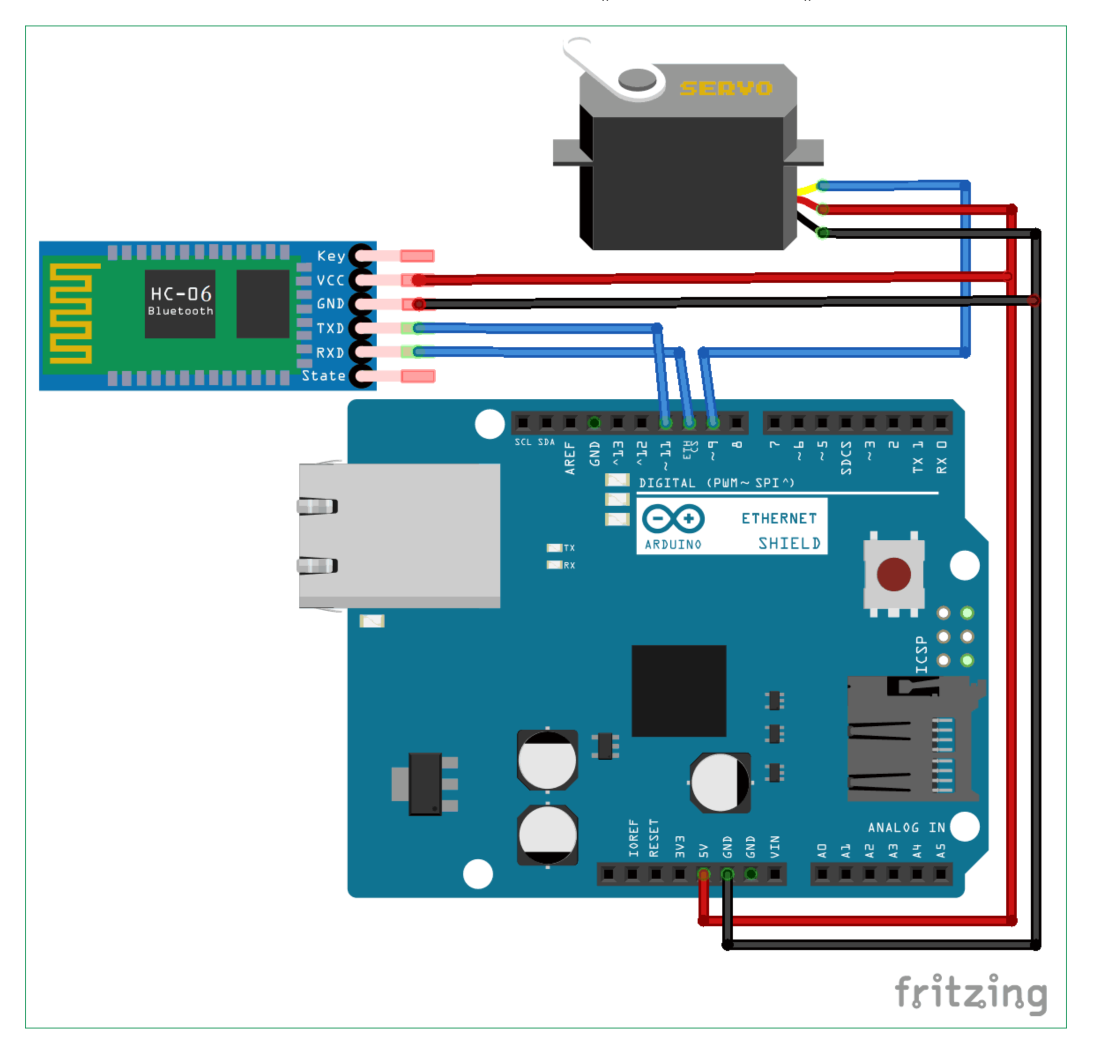


التحكم في محرك السيرفو باستخدام البلوتوث

في هذا البرنامج البرنامج سوف نوضح لك كيفية استخدام محرك سيرفو لاسلكيًا مع Arduino UNO وجهاز Android عبر اتصال Bluetooth. لقد قمنا بالفعل بالتحكم في محركات servo مع Arduino بواسطة الكبيل ، لكننا في هذه المرة نسيطر على Servo لاسلكيًا باستخدام القطعة Bluetooth HC-06.

تعمل هذه على إمدادات طاقة V5 وتعمل دبابيس الإشارة على V3.3 ، وبالتالي يكون منظم V3.3 موجودًا في الوحدة نفسها، لا داعي للقلق بشأنها. من بين ستة مسامير ستستخدم أربعة فقط في وضع التشغيل.

قم بتركيب الدائرة كما في المخطط التالي :

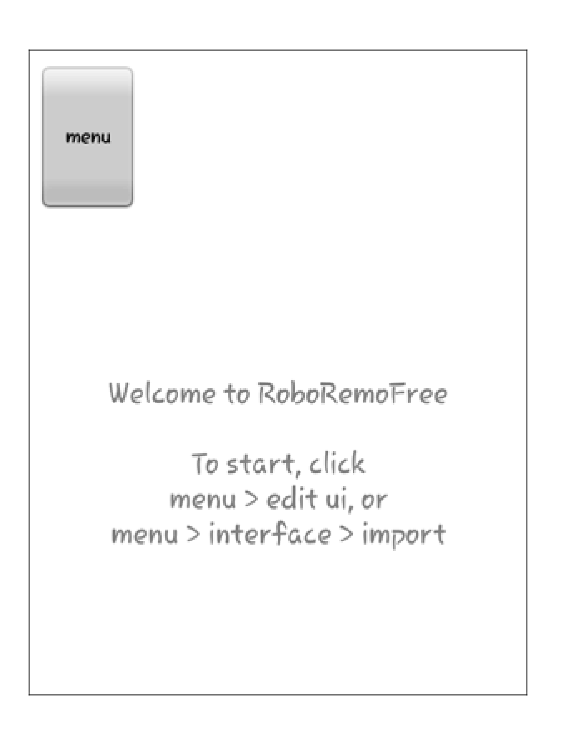


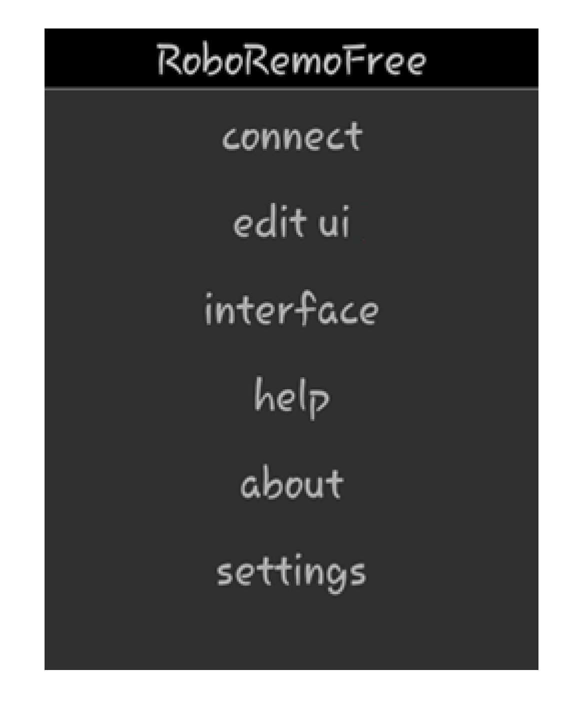
سوف نستخدم تطبيق Roboremo للتحكم في محرك ال Servo:

الخطوة 1: - قم بتنزيل تطبيق Roboremo من متجر Android Play وتثبيته في هاتفك الذكي.

https://play.google.com/store/apps/details?id=com.hardcodedjoy.roboremofree

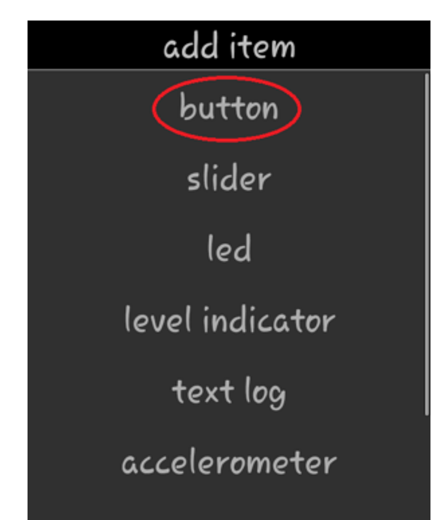
بعد تثبيته سترى نافذة التطبيق كما هو موضح في الشكل ، وبالضغط على زر "Menu" سترى النافذة الموضحة في الشكل الداكن أدناه:



الخطوة 2: - انقر على زر الاتصال connect وسترى النافذة الموضحة في الشكل أدناه ، ثم يجب عليك اختيار "بلوتوث RFCOMM" ومن ثم ستتمكن من توصيل وحدة HC-06 bluetooth مع تطبيق الروبوت الخاص بك "Roboremo".

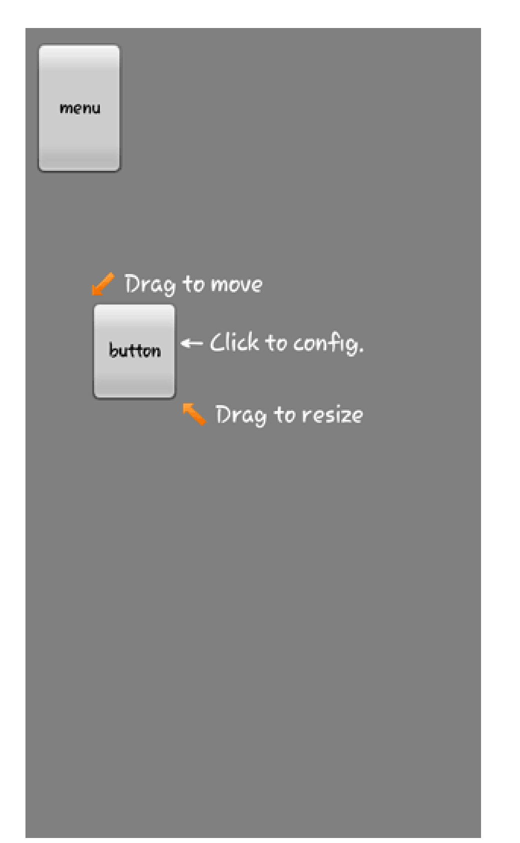
الخطوة 3: - بعد الاتصال بوحدة Bluetooth HC-06 عد إلى النافذة السابقة ثم انقر فوق "edit ui" لإنشاء واجهة المستخدم وفقًا لحاجتك.




عند النقر على "edit ui " سترى التطبيق مثل الشكل التالي واختر "Button " للحصول على هيكل الزر Button.

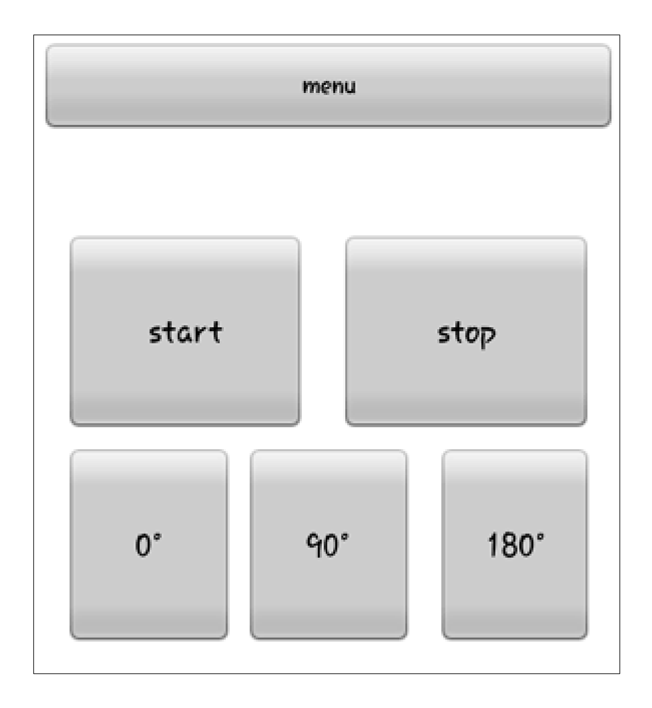
الخطوة 4: - بعد اختيار Button سوف تحصل على هيكل Button على الشاشة لتحريرها. يمكنك تغيير حجم الهيكل ونقله إلى أي مكان على الشاشة. الآن لتعيين القيمة التي يتم إرسالها عند النقر عبر البلوتوث ، عليك 'set press action' كما هو موضح في الشكل 6 وكتابة القيمة التي تريد إرسالها من هذا الزر المعين. مثل ، نرسل "1" لتدوير السيرفو عن طريق الضغط على زر "Start " في تطبيق Roboremo android.

يمكنك التحقق من جميع القيم ، التي يتم إرسالها عند النقر على أزرار مختلفة ، في الجدول المعطى في القسم التالي:






الخطوة 5: - وأخيرا لدينا واجهة المستخدم للتحكم في محرك سيرفو باستخدام الهاتف الذكي.




الان قم بتحميل الكود التالي :

```
#include <SoftwareSerial.h>

#include <Servo.h>


Servo myServo;

int TxD = 11;

int RxD = 10;

int servoposition;

int servopos;

int new1;

SoftwareSerial bluetooth(TxD, RxD);


void setup() {
```

```
  int pos=0;
  myServo.attach(9);
  myServo.write(0);
  Serial.begin(9600);       // start serial communication at 9600bps
  bluetooth.begin(9600);
}

void loop() {
 if (bluetooth.available())
   {
    String value = bluetooth.readString();
    servoposition = value.toInt();

    if (value.toInt() == 0)
    {
      Serial.println(servoposition);
    myServo.write(0);
      }

      if (value.toInt() == 45)
    {
      Serial.println(servoposition);
    myServo.write(45);
      }

      if (value.toInt() == 90)
```

```
 {
   Serial.println(servoposition);
 myServo.write(90);
   }

   if (value.toInt() == 135)
 {
   Serial.println(servoposition);
 myServo.write(135);
   }

   if (value.toInt() == 180)
 {
   Serial.println(servoposition);
 myServo.write(180);
   }

while(value.toInt()==1){
 if (bluetooth.available())
{
 value = bluetooth.readString();
 Serial.println(value);
 if (value.toInt()==2)
 {Serial.println("YYY");  break; }

}
```

```
    servopos++;

    delay(30);

    Serial.println(servopos);

    myServo.write(servopos);


    if (servopos ==180 )

    {servopos=0;break;}


  }

 }

}
```

في هذا المشروع ، نحن نسيطر على محرك سيرفو باستخدام تطبيق Android "Roboremo". في واجهة هذا التطبيق ، قمنا بإنشاء 5 أزرار للتحكم في محرك سيرفو. و عمل كل زر في الجدول أدناه:

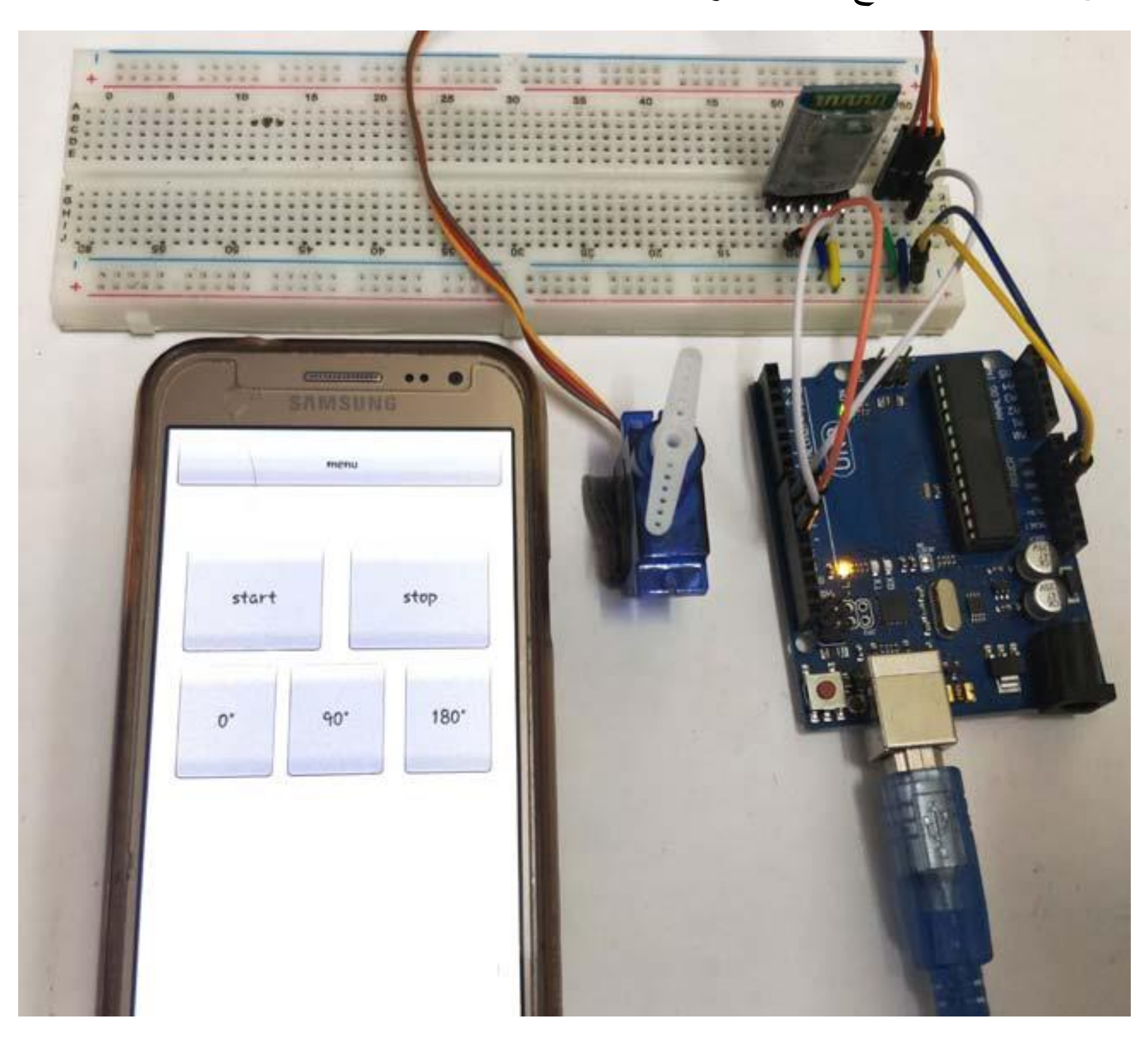
| S.No. | Button Name | Sending Value | Description |
|---|---|---|---|
| 1. | Start | 1 | This button is used to start rotating the servo from $0^0$ to $180^0$. |
| 2. | Stop | 2 | This button is used to stop the servo at any point. |
| 3. | $0^0$ | 0 | This button is used to rotate the servo to $0^0$. |
| 4. | $90^0$ | 90 | This button is used to rotate the servo to $90^0$. |
| 5. | $180^0$ | 180 | This button is used to rotate the servo to $180^0$. |

لذلك ، من خلال الضغط على هذه الأزرار على تطبيق الخاص بك Roboremo ، سيتم إرسال البيانات من خلال Bluetooth إلى وحدة بلوتوث HC-06. ثم استلام بياناتك بواسطة الوحدة النمطية HC-06 بواسطة Arduino و Arduino ، يتم تدوير Servo بزاوية محددة في الرمز الخاص بالزر الخاص. لقد تم ترميزنا أيضًا للزاوية

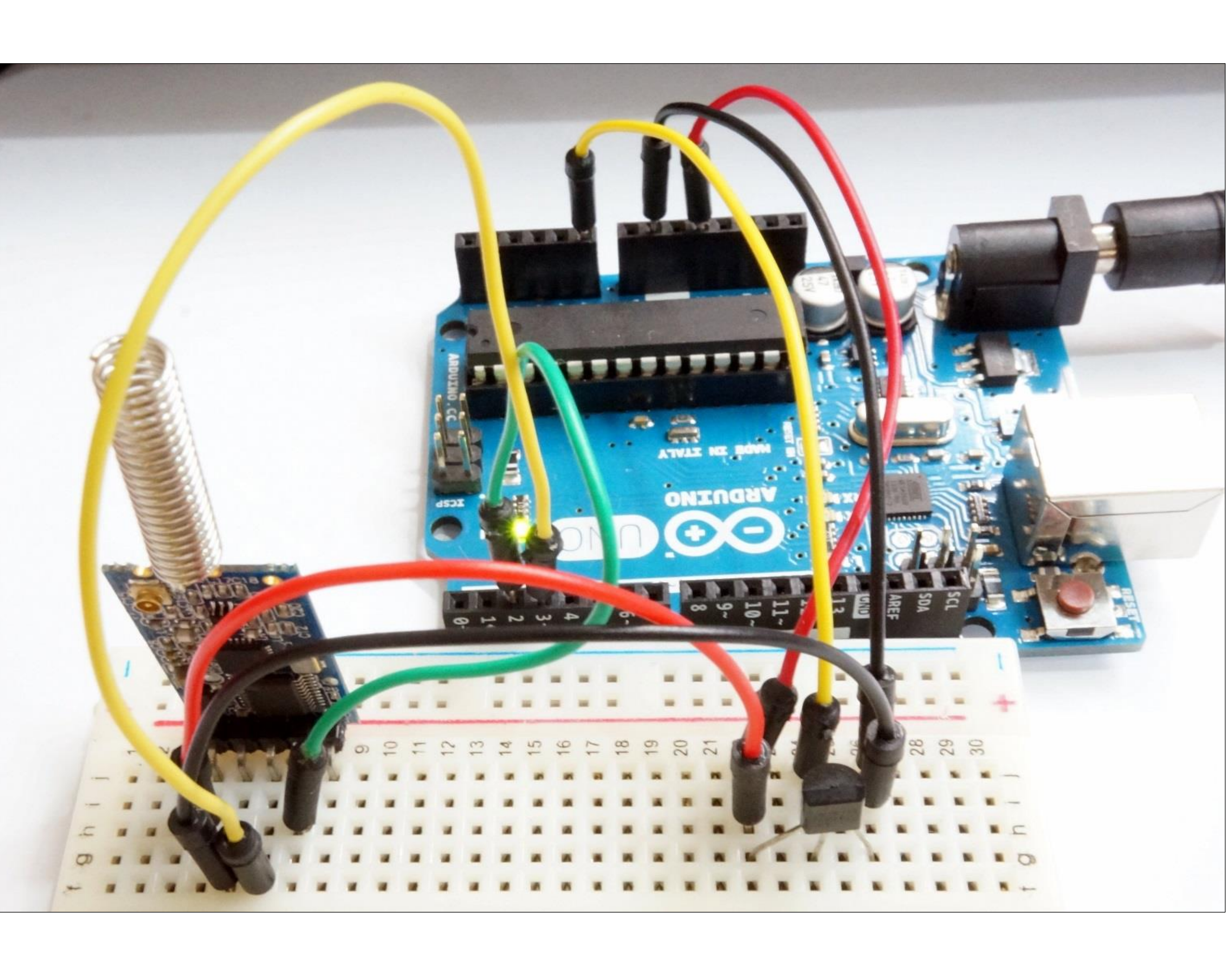
45 و 135 ، ولكن نظرًا لتقييد تطبيق Roboremo ، يمكنك فقط إنشاء 5 أزرار ، لذلك تخطينا زرين.

هذه هي الطريقة التي يمكنك بها إرسال البيانات من الهاتف الذكي إلى Arduino باستخدام البلوتوث للتحكم في الأجهزة لاسلكيًا. يمكن استخدام هذا في العديد من التطبيقات المتوفرة مجانا .

الان يمكنك الاستمتاع بهذه التجربة



- ثالثا : التحكم عن بعد باستخدام الوايرلس Wireless



في هذا الدرس ، سنتعلم كيفية استخدام وحدة الاتصالات اللاسلكية NRF24L01 و HC-12 التي يمكنها إجراء اتصال لاسلكي بعيد المدى بين لوحات Arduino المتعددة مع مسافات تصل إلى 1.8 كيلومتر.

## وحدة الاتصالات اللاسلكية طويلة المدى HC-12

في هذا الدرس ، قمنا بتقديم مثالين أساسيين يوضحان كيفية توصيل وحدة HC-12 وإجراء اتصال أساسي بين اثنين من لوحة ال Arduino ومثال إضافي حيث استخدم أداة استشعار التسارع accelerometer sensor مع اردوينوالاول للتحكم لاسلكيًا في موضع stepper في اللوحة الثانية من الاردوينو.




أولاً ، دعنا نلقي نظرة على وحدة اتصالات المنفذ التسلسلي اللاسلكي HC-12. وهنا بعض المواصفات:

- يتراوح نطاق تردد العمل اللاسلكي من 433.4 ميغاهرتز إلى 473.0 ميغاهيرتز
- لديها ما مجموعه 100 قناة مع خطوة stepping 400 كيلوهرتز بين كل قناة
- قوة الإرسال from -1dBm (0.79mW) to 20dBm (100mW)
- حساسية الاستقبال from -117dBm (0.019pW) to -100dBm (10pW)

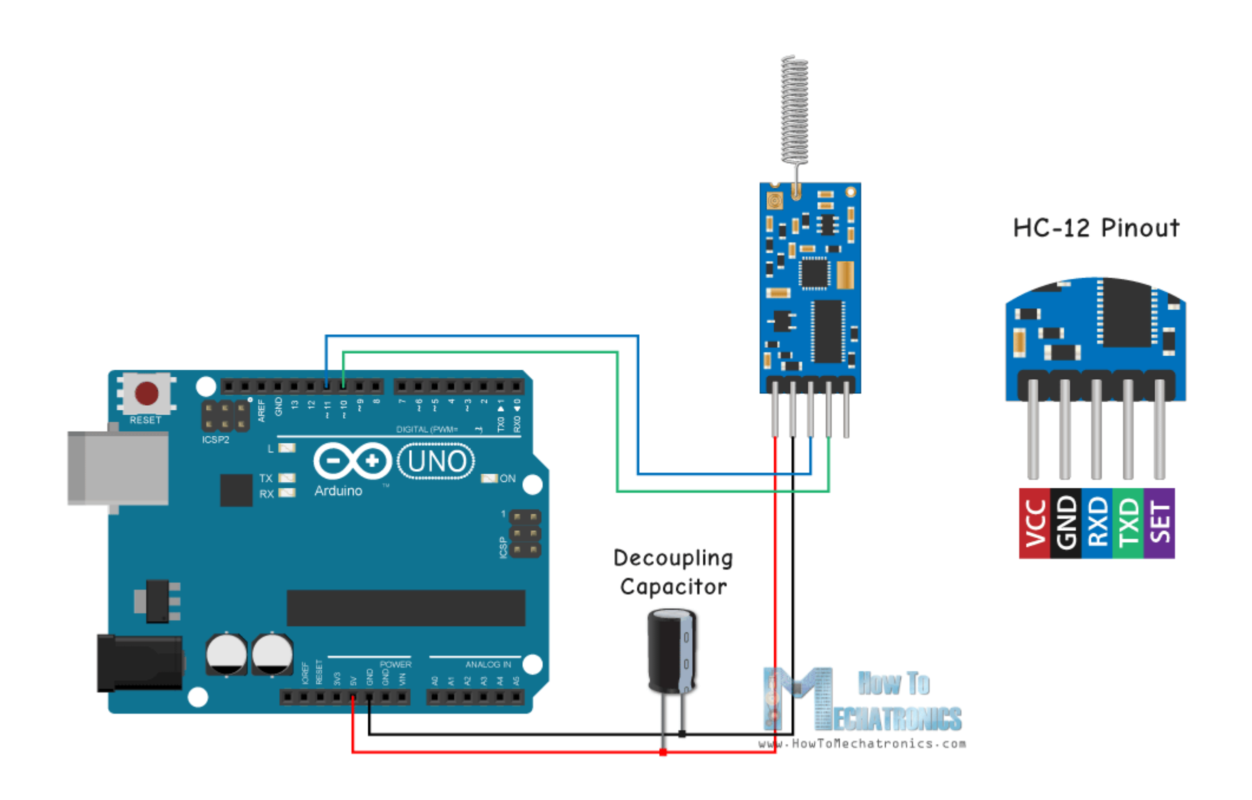
تعتمد هذه القيم فعليًا على معدل البث التسلسلي والإضافي Serial and Over-the-Air Baud المحدد كما هو موضح في الجدول :

| Receiver Sensitivity | Over-the-Air Baud Rate | Serial Port Baud Rate |
|---|---|---|
| -117 dBm | 5000 bps | 1200 bps |
| -117 dBm | 5000 bps | 2400 bps |
| -112 dBm | 15000 bps | 4800 bps |
| -112 dBm | 15000 bps | 9600 bps |
| -107 dBm | 58000 bps | 19200 bps |
| -107 dBm | 58000 bps | 38400 bps |
| -100 dBm | 236000 bps | 57600 bps |
| -100 dBm | 236000 bps | 115200 bps |

تحتوي وحدة HC-12 على متحكم لا يحتاج المستخدم إلى برمجته بالفعل , لاعدادات تكوين الوحدة ، نستخدم أوامر AT ، والتي يمكن إرسالها من Arduino أو كمبيوتر شخصي أو أي متحكم آخر باستخدام المنفذ التسلسلي serial port. للدخول إلى وضع أوامر AT ، يتعين علينا فقط تعيين دبوس "Set" في الوحدة النمطية على مستوى منطقي منخفض low logic level .

الآن دعنا نربط وحدة HC-12 بأردوينو ونقدم المثال الأول. إليكم مخططات الدوائر. يتراوح الجهد التشغيلي للوحدة من 3.2 فولت إلى 5.5 فولت ، ويوصى باستخدام مكثف فصل ومصدر طاقة خارجي للحصول على عمل أكثر ثباتًا. ومع ذلك ، استخدمت الكمبيوتر الشخصي USB كمصدر تغذية لكل الأمثلة الثلاثة في هذا الدرس التعليمي ولم أواجه أي مشكلة في ذلك.

لقد قمت بتوصيل الوحدة الأولى بـ Arduino UNO والوحدة الثانية بـ Arduino MEGA ، لكن بالطبع يمكنك استخدام أي لوحة تريدها.




إليك الكود للمثال الأول ، وهو اتصال أساسي بين الوحدتين باستخدام Serial Monitor :

```
/*    Arduino Long Range Wireless Communication using HC-12

                       Example 01

   by Dejan Nedelkovski, www.HowToMechatronics.com

*/


#include <SoftwareSerial.h>


SoftwareSerial HC12(10, 11); // HC-12 TX Pin, HC-12 RX Pin


void setup() {

  Serial.begin(9600);             // Serial port to computer
```

```
  HC12.begin(9600);               // Serial port to HC12


}


void loop() {

  while (HC12.available()) {        // If HC-12 has data

    Serial.write(HC12.read());      // Send the data to Serial monitor

  }

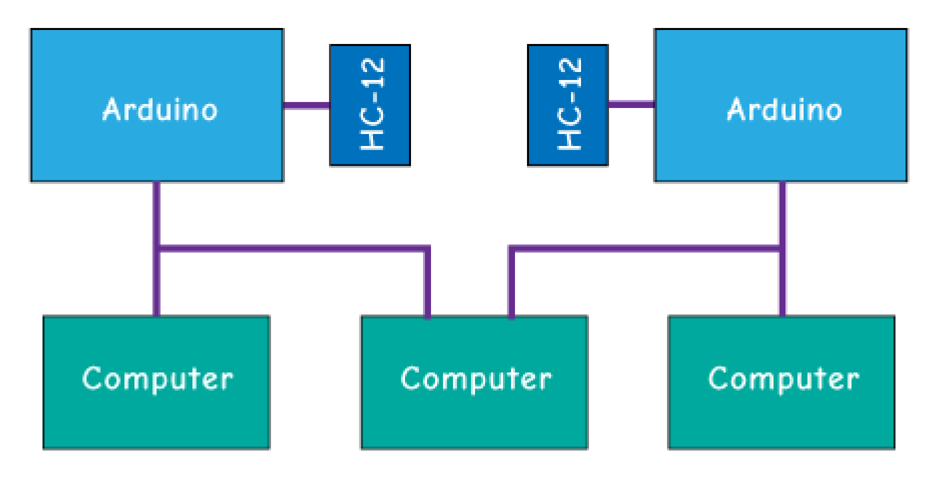
  while (Serial.available()) {      // If Serial monitor has data

    HC12.write(Serial.read());      // Send that data to HC-12

  }

}
```

يتم استخدام نفس الكود لكل من الاردوينو. يمكننا توصيل جهازي Arduinos على جهازي كمبيوتر منفصلين ولكن يمكننا أيضًا استخدام جهاز كمبيوتر واحد.




في هذه الحالة ، بمجرد توصيل أول Arduino بالكمبيوتر ، نحتاج إلى تحديد النموذج ومنفذ COM وتحميل الرمز إلى Arduino. بعد ذلك نقوم بتوصيل Arduino الثاني وعلينا أن نبدأ تشغيل Arduino IDE مرة أخرى حتى نتمكن من تحديد منفذ COM الآخر الذي يتصل به Arduino الثاني ، ثم نرفع نفس الكود.

لذلك بمجرد تشغيل IDE الخاصين بـ Arduino ، يمكننا تشغيل الشاشات التسلسلية واختبار ما إذا كان الاتصال يعمل بشكل صحيح. سيتم إرسال أي شيء نكتبه في الشاشة التسلسلية من واحدة إلى أخرى من Arduino :



كيف يعمل البرنامج : إذن بمجرد كتابة شيء ما في الشاشة التسلسلية والنقر فوق الزر "إرسال" ، في أول Arduino ، ستصبح حلقة التكرار مع وظيفة Serial.available () صحيحة وستستخدم الدالة HC12.write () إرسال البيانات من الشاشة التسلسلية إلى وحدة HC-12. ستقوم هذه الوحدة بنقل البيانات لاسلكيا إلى الوحدة HC-12 الثانية ، لذلك في حلقة اردوينو الثانية ، ستصبح الحلقة أثناء استخدام وظيفة HC12.available () صحيحة ، وباستخدام وظيفة Serial.write () ، سيتم إرسال البيانات إلى رصد المتسلسل.

يمكننا استخدام نفس الكود لإرسال أوامر AT وتكوين اوامر جديدة في الوحدة. كل ما يتعين علينا القيام به هو توصيل دبوس "Set" من الوحدة إلى Ground أو أي دبوس رقمي من Arduino وضبط الدبوس على مستوى منطقي منخفض.

لاختبار ما إذا كنا قد دخلنا في الوضع بنجاح ، في الشاشة التسلسلية يمكننا كتابة "AT" ويجب أن نحصل على رسالة استجابة "OK". يوجد إجمالي 12 أوامر AT ، ويتم استخدامها لتغيير المعلمات المختلفة مثل معدل البث بالباود والقناة وقدرة الإرسال ، إلخ. على سبيل المثال ، إذا كتبنا "AT + B38400" ، فسيتم تعيين معدل الباود للوحدة النمطية على 38400.

امثلة على أوامر AT :

| | |
|---|---|
| AT – Test command | Example: Send "AT" to module, and the module returns "OK" |
| AT+Bxxxx – Change the serial port baud rate | Available baud rates: 1200 bps, 2400 bps, 4800 bps, 9600 bps, 19200 bps, 38400 bps, 57600 bps, and 115200 bps. Default: 9600 bps<br><br>Example: Send "AT+B38400" to module, and the module returns "OK+B19200". |
| AT+Cxxxx – Change wireless communication channel, from 001 to 100 | Default: Channel 001, with working frequency of 433.4MHz. Each next channel is 400KHz higher.<br><br>Example: If we want to set the module to channel 006, we need to send "AT+C006" command to the module, and the module will return "OK+C006". The new working frequency will be 435.4MHz |

الآن دعنا ننقل المثال الثاني. هنا سنستخدم زري ضغط لاختيار قنوات اتصال مختلفة ومشاهدة طريقة مختلفة لتخزين البيانات الواردة.

ملاحظة: يتم توصيل دبابيس "Set" في كلتا وحدتي HC-12 بالدبابيس رقم 6 من Arduinos واثنين من الأزرار ، في Arduino الأول ، إلى الطرفين 4 و 3.

كود الاردوينو الاول :

```
/*    Arduino Long Range Wireless Communication using HC-12

    Example 02 - Changing channels using push buttons - Buttons side

   by Dejan Nedelkovski, www.HowToMechatronics.com

*/
```

```
#include <SoftwareSerial.h>

#define setPin 6
#define button1 4
#define button2 3

SoftwareSerial HC12(10, 11);         // HC-12 TX Pin, HC-12 RX Pin

byte incomingByte;
String readBuffer = "";

int button1State = 0;
int button1Pressed = 0;
int button2State = 0;
int button2Pressed = 0;

void setup() {
  Serial.begin(9600);                  // Open serial port to computer
  HC12.begin(9600);                    // Open serial port to HC12
  pinMode(setPin, OUTPUT);
  pinMode(button1, INPUT);
  pinMode(button2, INPUT);
  digitalWrite(setPin, HIGH);          // HC-12 normal, transparent mode
}
```

```
void loop() {

  // ==== Storing the incoming data into a String variable

  while (HC12.available()) {             // If HC-12 has data

    incomingByte = HC12.read();          // Store each icoming byte from HC-
12

    readBuffer += char(incomingByte);    // Add each byte to ReadBuffer
string variable

  }

  delay(100);

  // ==== Sending data from one HC-12 to another via the Serial Monitor

  while (Serial.available()) {

    HC12.write(Serial.read());

  }



  // ==== If button 1 is pressed, set the channel 01

  button1State = digitalRead(button1);

  if (button1State == HIGH & button1Pressed == LOW) {

    button1Pressed = HIGH;

    delay(20);

  }

  if (button1Pressed == HIGH) {

    HC12.print("AT+C001");               // Send the AT Command to the other
module

    delay(100);

    //Set AT Command Mode

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(100);                          // Wait for the HC-12 to enter AT
Command mode
```

```
    HC12.print("AT+C001");               // Send AT Command to HC-12

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());         // Send the data to Serial monitor

    }

    Serial.println("Channel successfully changed");

    digitalWrite(setPin, HIGH);          // Exit AT Command mode

    button1Pressed = LOW;

  }



  // ====  If button 2 is pressed, set the channel 02

  button2State = digitalRead(button2);

  if (button2State == HIGH & button2Pressed == LOW) {

    button2Pressed = HIGH;

    delay(100);

  }

  if (button2Pressed == HIGH) {

    HC12.print("AT+C002"); // Send the AT Command to the other module

    delay(100);

    //Set AT Command Mode

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(100);                          // Wait for the HC-12 to enter AT
Command mode

    HC12.print("AT+C002");               // Send AT Command to HC-12

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command
response)
```

```
      Serial.write(HC12.read());         // Send the data to Serial monitor

    }

    Serial.println("Channel successfully changed");

    digitalWrite(setPin, HIGH);

    button2Pressed = LOW;

  }

  checkATCommand();

  readBuffer = "";                       // Clear readBuffer

}

// ==== Custom function - Check whether we have received an AT Command via
the Serial Monitor

void checkATCommand () {

  if (readBuffer.startsWith("AT")) {     // Check whether the String starts
with "AT"

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(200);                          // Wait for the HC-12 to enter AT
Command mode

    HC12.print(readBuffer);              // Send AT Command to HC-12

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());         // Send the data to Serial monitor

    }

    digitalWrite(setPin, HIGH);          // Exit AT Command mode

  }

}
```

كود الاردوينو الثاني :

```
/*    Arduino Long Range Wireless Communication using HC-12

        Example 02 - Changing channels using push buttons

   by Dejan Nedelkovski, www.HowToMechatronics.com

*/


#include <SoftwareSerial.h>


#define setPin 6


SoftwareSerial HC12(10, 11); // HC-12 TX Pin, HC-12 RX Pin


byte incomingByte;

String readBuffer = "";


void setup() {

  Serial.begin(9600);             // Open serial port to computer

  HC12.begin(9600);               // Open serial port to HC12

  pinMode(setPin, OUTPUT);

  digitalWrite(setPin, HIGH);     // HC-12 normal mode

}


void loop() {

  // ==== Storing the incoming data into a String variable

  while (HC12.available()) {             // If HC-12 has data
```

```
    incomingByte = HC12.read();          // Store each icoming byte from HC-
12

    readBuffer += char(incomingByte);    // Add each byte to ReadBuffer
string variable

  }

  delay(100);

  // ==== Sending data from one HC-12 to another via the Serial Monitor

  while (Serial.available()) {

    HC12.write(Serial.read());

  }

  // === If button 1 is pressed, set channel 01

  if (readBuffer == "AT+C001") {

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(100);                          // Wait for the HC-12 to enter AT
Command mode

    HC12.print(readBuffer);              // Send AT Command to HC-12
("AT+C001")

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());         // Send the data to Serial monitor

    }

    Serial.println("Channel successfully changed");

    digitalWrite(setPin, HIGH);          // Exit AT Command mode

    readBuffer = "";

  }

  // === If button 2 is pressed, set channel 02

  if (readBuffer == "AT+C002") {

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode
```

```
    delay(100);                          // Wait for the HC-12 to enter AT
Command mode

    HC12.print(readBuffer);              // Send AT Command to HC-12

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());         // Send the data to Serial monitor

    }

    Serial.println("Channel successfully changed");

    digitalWrite(setPin, HIGH);          // Exit AT Command mode



    readBuffer = "";

  }

  checkATCommand();

  readBuffer = "";                       // Clear readBuffer

}

// ==== Custom function - Check whether we have received an AT Command via
the Serial Monitor

void checkATCommand () {

  if (readBuffer.startsWith("AT")) {     // Check whether the String starts
with "AT"

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(100);                          // Wait for the HC-12 to enter AT
Command mode

    HC12.print(readBuffer);              // Send AT Command to HC-12

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());         // Send the data to Serial monitor
```

```
    }

    digitalWrite(setPin, HIGH);          // Exit AT Command mode

  }

}
```

شرح الكود : نحتاج أولاً إلى تحديد المسامير وتعيين دبوس "set" على مستوى منطقي مرتفع حتى تعمل الوحدة النمطية في الوضع العادي والشفاف normal, transparent . من خلال الحلقة الأولى ، نقوم بتخزين البيانات الواردة في متغير String ، حتى نتمكن من التعامل معها بشكل أفضل :

```
// ==== Storing the incoming data into a String variable

  while (HC12.available()) {             // If HC-12 has data

    incomingByte = HC12.read();          // Store each icoming byte from HC-
12

    readBuffer += char(incomingByte);    // Add each byte to ReadBuffer
string variable

  }
```

تأتي البيانات الواردة دائمًا بايت واحد في المرة الواحدة ، لذلك على سبيل المثال ، إذا أرسلنا السلسلة "Test123" من Arduino الثانية ، فستعمل هذه الحلقة على 7 تكرارات. كل تكرار باستخدام دالة HC12.read () سوف نقرأ كل بايت أو حرف وارد وإضافته إلى متغير String المسمى "readBuffer".

Example: "Test123"

| Iteration 1: | Iteration 2: | Iteration 3: | Iteration 4: | Iteration 5: | Iteration 6: | Iteration 7: |
|---|---|---|---|---|---|---|
| incomingByte: **T** | incomingByte: **e** | incomingByte: **s** | incomingByte: **t** | incomingByte: **1** | incomingByte: **2** | incomingByte: **3** |
| readBuffer: **T** | readBuffer: **Te** | readBuffer: **Tes** | readBuffer: **Test** | readBuffer: **Test1** | readBuffer: **Test12** | readBuffer: **Test123** |

بعد ذلك ، دعنا نرى كيف يمكننا تغيير قناة الاتصال باستخدام زر الضغط الأول. لذلك إذا ضغطنا على زر الضغط الأول ، باستخدام وظيفة HC12.print () ، فسوف نرسل string "AT + C001" إلى الوحدة النمطية HC-12 أو إلى وحدة Arduino الثانية.

```
if (button1Pressed == HIGH) {

    HC12.print("AT+C001");               // Send the AT Command to the other
module

    delay(100);

    //Set AT Command Mode

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(100);                          // Wait for the HC-12 to enter AT
Command mode

    HC12.print("AT+C001");               // Send AT Command to HC-12

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());         // Send the data to Serial monitor

    }

    Serial.println("Channel successfully changed");

    digitalWrite(setPin, HIGH);          // Exit AT Command mode

    button1Pressed = LOW;

  }
```

عندما يتم استلام هذه string في Arduino الثاني ، سنقوم بتعيين الوحدة HC-12 في وضع أوامر AT ، ثم نكتب string نفسها "AT + C001" التي ستقوم بتعيين الوحدة النمطية على قناة الاتصال رقم واحد.

```
// At the second Arduino



// === If button 1 is pressed, set channel 01

  if (readBuffer == "AT+C001") {

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(100);                          // Wait for the HC-12 to enter AT
Command mode
```

```
    HC12.print(readBuffer);               // Send AT Command to HC-12
("AT+C001")

    delay(200);

    while (HC12.available()) {            // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());          // Send the data to Serial monitor

    }

    Serial.println("Channel successfully changed");

    digitalWrite(setPin, HIGH);           // Exit AT Command mode

    readBuffer = "";

  }
```

نستخدم الحلقة التالية أثناء طباعة رسالة الاستجابة من وحدة HC-12 بينما تم تغيير القناة بنجاح.

```
while (HC12.available()) {            // If HC-12 has data (the AT Command
response)

      Serial.write(HC12.read());          // Send the data to Serial monitor

    }
```

مرة أخرى في Arduino الأول ، نحن نقوم بنفس الإجراء لإرسال أمر AT إلى الوحدة النمطية HC-12 الأولى. بنفس الطريقة ، باستخدام الضغط على الزر الثاني ، قمنا بتعيين قناة الاتصال رقم اثنين. لذلك باستخدام هذه الطريقة ، يمكننا في أي وقت اختيار وحدة HC-12 التي سنتواصل معها.

في النهاية ، تقوم وظيفة checkCommand () المخصصة ، بالتحقق مما إذا كانت الرسالة المستلمة هي أمر AT ، عن طريق التحقق مما إذا كانت string تبدأ بـ "AT". إذا كان الأمر كذلك ، تدخل الوحدة النمطية في وضع أوامر AT وتنفذ الأمر.

```
// ==== Custom function - Check whether we have received an AT Command via
the Serial Monitor

void checkATCommand () {
```

```
  if (readBuffer.startsWith("AT")) {     // Check whether the String starts with "AT"

    digitalWrite(setPin, LOW);           // Set HC-12 into AT Command mode

    delay(200);                          // Wait for the HC-12 to enter AT Command mode

    HC12.print(readBuffer);              // Send AT Command to HC-12

    delay(200);

    while (HC12.available()) {           // If HC-12 has data (the AT Command response)

      Serial.write(HC12.read());         // Send the data to Serial monitor

    }

    digitalWrite(setPin, HIGH);          // Exit AT Command mode

  }

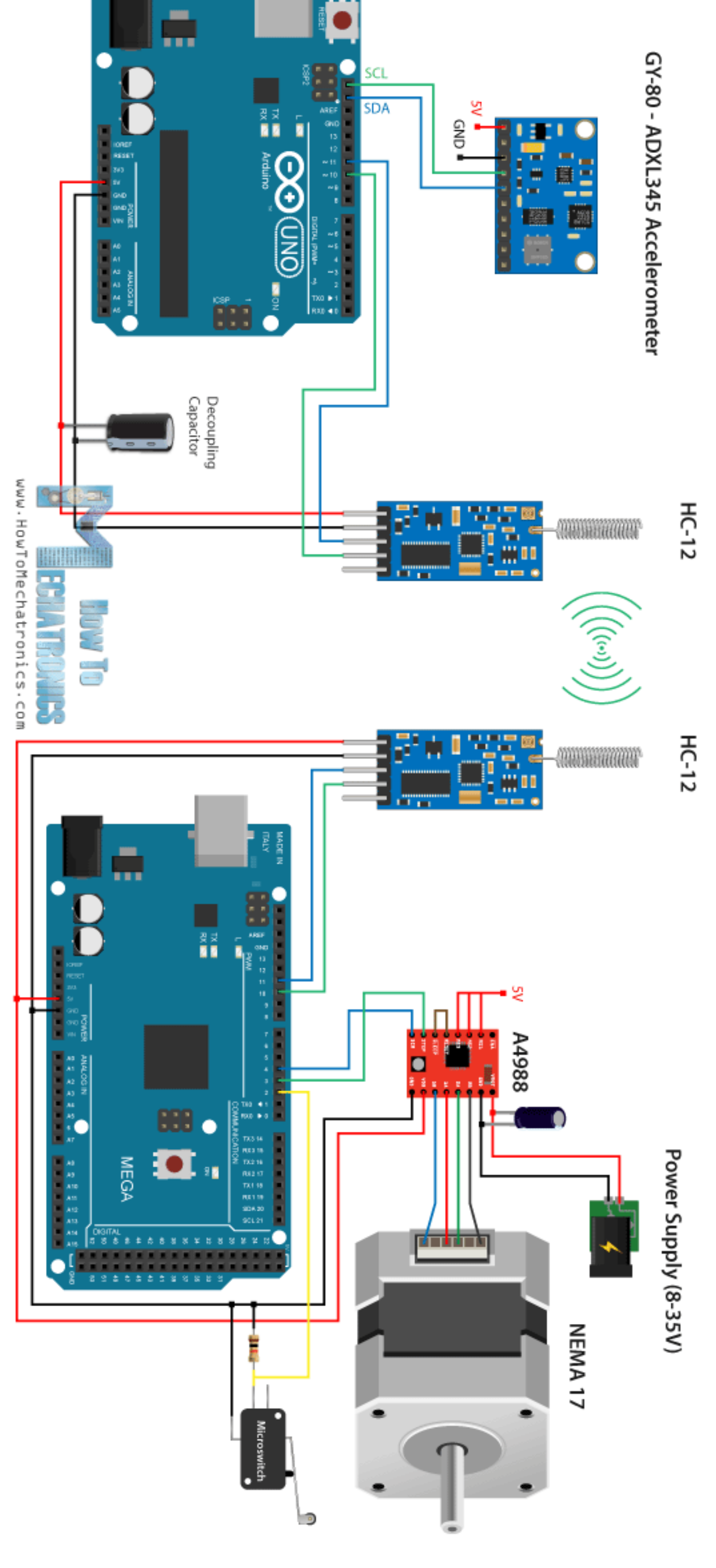
}
```

التحكم في المحركات الخطوية Stepper Motor باستخدام مقياس التسارع Accelerometer

الآن دعونا نلقي نظرة على المثال الثالث , نحن هنا نتحكم في موضع المحرك الخطوي في Arduino الثاني ، باستخدام وحدة التسارع في Arduino الأول.

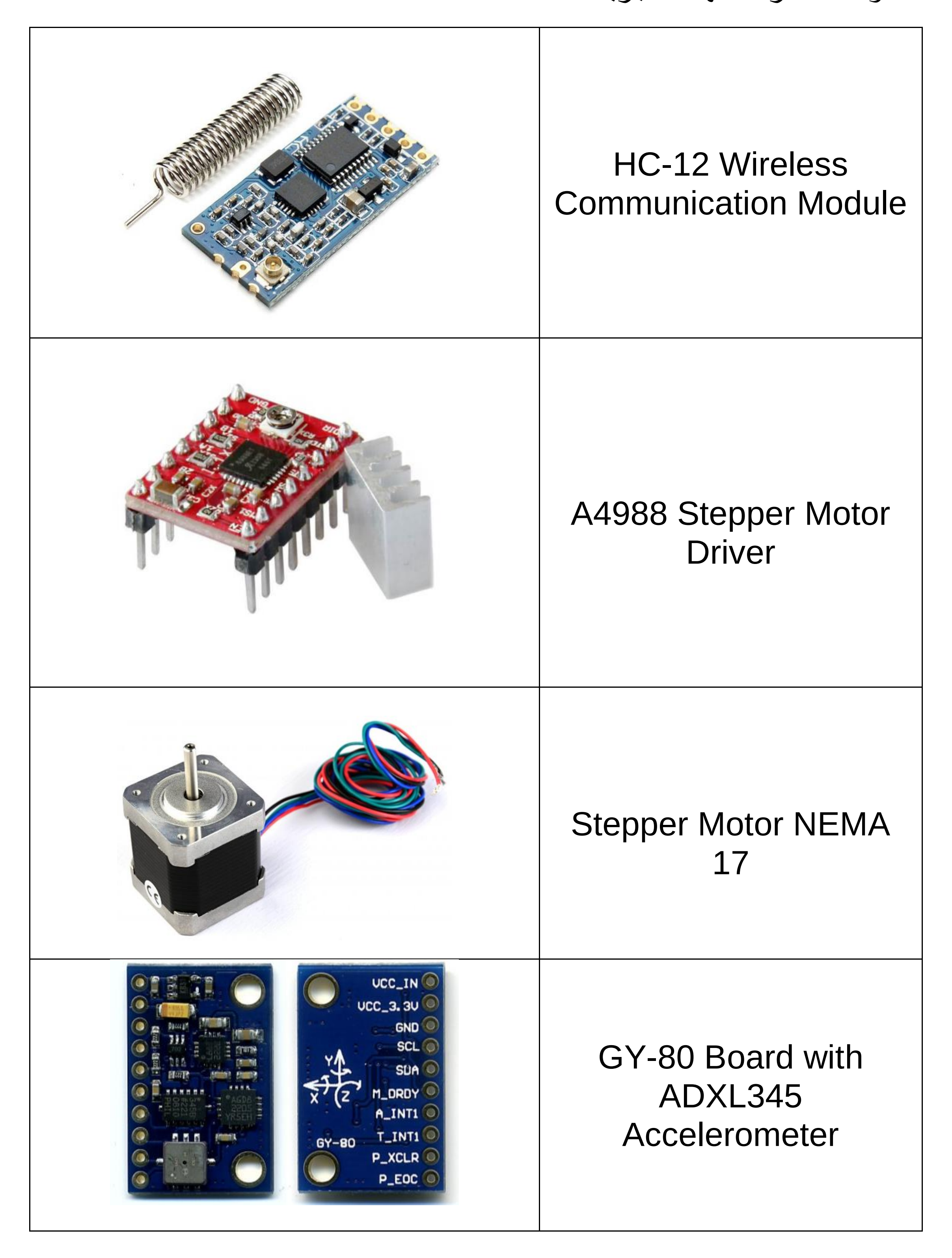
تحتوي الدائرة أيضًا على microswitch للعثور على الوضع الأولي للمحرك الخطوي عند 0 درجة.

مخطط التجربة :



HC-12 Wireless Communication: Stepper Motor Control using an Accelerometer
GY-80 - ADXL345 Accelerometer
5V
GND
SCL
SDA
HC-12
HC-12
Decoupling Capacitor
www.HowToMechatronics.com
A4988
5V
Power Supply (8-35V)
NEMA 17
Microswitch
MEGA
UNO

المكونات اللازمة لهذه التجربة :




| | |
|---|---|
| | HC-12 Wireless Communication Module |
| | A4988 Stepper Motor Driver |
| | Stepper Motor NEMA 17 |
| | GY-80 Board with ADXL345 Accelerometer |

الاردوينو الاول - كود الإرسال :

```
/*   Arduino Long Range Wireless Communication using HC-12

     Example 03 - Stepper Motor Control using Accelerometer - Transmitter, Accelerometer

    by Dejan Nedelkovski, www.HowToMechatronics.com

*/



#include <SoftwareSerial.h>

#include <Wire.h>



SoftwareSerial HC12(10, 11); // HC-12 TX Pin, HC-12 RX Pin



float angle;

int lastAngle = 0;

int count = 0;

int angleSum = 0;



//--- Accelerometer Register Addresses

#define Power_Register 0x2D

#define X_Axis_Register_DATAX0 0x32 // Hexadecima address for the DATAX0 internal register.

#define X_Axis_Register_DATAX1 0x33 // Hexadecima address for the DATAX1 internal register.

#define Y_Axis_Register_DATAY0 0x34

#define Y_Axis_Register_DATAY1 0x35

#define Z_Axis_Register_DATAZ0 0x36

#define Z_Axis_Register_DATAZ1 0x37
```

```
int ADXAddress = 0x53;  //Device address in which is also included the 8th
bit for selecting the mode, read in this case.

int X0, X1, X_out;

int Y0, Y1, Y_out;

int Z1, Z0, Z_out;

float Xa, Ya, Za;



void setup() {

  HC12.begin(9600);               // Open serial port to HC12

  Wire.begin(); // Initiate the Wire library

  Serial.begin(9600);

  delay(100);



  Wire.beginTransmission(ADXAddress);

  Wire.write(Power_Register); // Power_CTL Register

  // Enable measurement

  Wire.write(8); // Bit D3 High for measuring enable (0000 1000)

  Wire.endTransmission();

}

void loop() {

  // X-axis

  Wire.beginTransmission(ADXAddress); // Begin transmission to the Sensor

  //Ask the particular registers for data

  Wire.write(X_Axis_Register_DATAX0);

  Wire.write(X_Axis_Register_DATAX1);

  Wire.endTransmission(); // Ends the transmission and transmits the data
from the two registers
```

```
  Wire.requestFrom(ADXAddress, 2); // Request the transmitted two bytes from the two registers

  if (Wire.available() <= 2) { //

    X0 = Wire.read(); // Reads the data from the register

    X1 = Wire.read();

    /* Converting the raw data of the X-Axis into X-Axis Acceleration

      - The output data is Two's complement

      - X0 as the least significant byte

      - X1 as the most significant byte */

    X1 = X1 << 8;

    X_out = X0 + X1;

    Xa = X_out / 256.0; // Xa = output value from -1 to +1, Gravity acceleration acting on the X-Axis

  }

  //Serial.print("Xa= ");

  //Serial.println(X_out);



  // Y-Axis

  Wire.beginTransmission(ADXAddress);

  Wire.write(Y_Axis_Register_DATAY0);

  Wire.write(Y_Axis_Register_DATAY1);

  Wire.endTransmission();

  Wire.requestFrom(ADXAddress, 2);

  if (Wire.available() <= 2) {

    Y0 = Wire.read();

    Y1 = Wire.read();

    Y1 = Y1 << 8;
```

```
    Y_out = Y0 + Y1;

    Ya = Y_out / 256.0;

  }


  // Combine X and Y values for getting the angle value from 0 to 180 degrees

  if (Y_out > 0) {

    angle = map(Y_out, 0, 256, 90, 0);

  }

  else if (Y_out < 0) {

    angle = map(Y_out, 256, 0, 90, 0);

    angle = 90 - angle;

  }

  if (X_out < 0 & Y_out < 0) {

    angle = 180;

  }

  if (X_out < 0 & Y_out >0) {

    angle = 0;

  }


  // float to int

  int angleInt = int(angle);

  // Makes 100 accelerometer readings and sends the average for smoother
result

  angleSum = angleSum + angleInt;

  count++;

  if (count >= 100) {

    angleInt = angleSum / 100;
```

```
    angleSum = 0;

    count = 0;

    // Some more smoothing of acceleromter reading - sends the new angle only
if it differes from the previous one by +-2

    if (angleInt > lastAngle + 2 || angleInt < lastAngle - 2) {

      Serial.println(angleInt);

      String angleString = String(angleInt);

      //sends the angle value with start marker "s" and end marker "e"

      HC12.print("s" + angleString + "e");

      delay(10);

      lastAngle = angleInt;

      angleSum = 0;

      count = 0;

    }

  }

}
```

الاردوينو الثاني - كود الاستقبال:

```
/*   Arduino Long Range Wireless Communication using HC-12

     Example 03 - Stepper Motor Control using Accelerometer - Receiver,
Stepper Motor



    by Dejan Nedelkovski, www.HowToMechatronics.com

*/

#include <SoftwareSerial.h>



SoftwareSerial HC12(10, 11); // HC-12 TX Pin, HC-12 RX Pin
```

```
char incomingByte;
String readBuffer = "";

// defines pins numbers
const int dirPin = 4;
const int stepPin = 3;
const int button = 2;

int currentAngle = 0;
int lastAngle = 0;
int rotate = 0;

void setup() {
  Serial.begin(9600);             // Open serial port to computer
  HC12.begin(9600);               // Open serial port to HC12

  // Sets the two pins as Outputs
  pinMode(dirPin, OUTPUT);
  pinMode(stepPin, OUTPUT);
  // Microswitch input, with internal pull-up resistor activated
  pinMode(button, INPUT_PULLUP);
  delay(10);
  digitalWrite(dirPin, HIGH);
  boolean startingPosition = true;
  while (startingPosition) {
```

```
    digitalWrite(stepPin, HIGH);

    delayMicroseconds(200);

    digitalWrite(stepPin, LOW);

    delayMicroseconds(200);

    if (digitalRead(button) == LOW) {

      startingPosition = false;

    }

  }

  delay(100);

}

void loop() {

  readBuffer = "";

  boolean start = false;

  // Reads the incoming angle

  while (HC12.available()) {             // If HC-12 has data

    incomingByte = HC12.read();          // Store each icoming byte from HC-
12

    delay(5);

    // Reads the data between the start "s" and end marker "e"

    if (start == true) {

      if (incomingByte != 'e') {

        readBuffer += char(incomingByte);    // Add each byte to ReadBuffer
string variable

      }

      else {

        start = false;

      }
```

```
    }
    // Checks whether the received message statrs with the start marker "s"
    else if ( incomingByte == 's') {
      start = true; // If true start reading the message
    }
  }
  // Converts the string into integer
  currentAngle = readBuffer.toInt();
  // Makes sure it uses angles between 0 and 180
  if (currentAngle > 0 && currentAngle < 180) {
    // Convert angle value to steps (depending on the selected step
resolution)
    // A cycle = 200 steps, 180deg = 100 steps ; Resolution: Sixteenth step
x16
    currentAngle = map(currentAngle, 0, 180, 0, 1600);
    //Serial.println(currentAngle); // Prints the angle on the serial monitor
    digitalWrite(dirPin, LOW); // Enables the motor to move in a particular
direction
    // Rotates the motor the amount of steps that differs from the previous
positon
    if (currentAngle != lastAngle) {
      if (currentAngle > lastAngle) {
        rotate = currentAngle - lastAngle;
        for (int x = 0; x < rotate; x++) {
          digitalWrite(stepPin, HIGH);
          delayMicroseconds(400);
          digitalWrite(stepPin, LOW);
          delayMicroseconds(400);
```

```
        }
      }
      // rotate the other way
      if (currentAngle < lastAngle) {
        rotate = lastAngle - currentAngle;
        digitalWrite(dirPin, HIGH);        //Changes the rotations direction
        for (int x = 0; x < rotate; x++) {
          digitalWrite(stepPin, HIGH);
          delayMicroseconds(400);
          digitalWrite(stepPin, LOW);
          delayMicroseconds(400);
        }
      }
    }
    lastAngle = currentAngle;  // Remembers the current/ last positon
  }
}
```

وصف الكود (الجزء المخصص للوايرلس) :

أولاً نحدد الاطراف pins ونهيئ الوحدات modules في قسم الإِعداد setup section . ثم نقرأ قيم المحور X و Y لمقياس التسارع ونقوم بتعيينها إلى قيم من 0 إلى 180 درجة. في بعض الأحيان قد تكون القيم القادمة من مقياس التسارع غير مستقرة أو تهتز ، لذلك ، ولتجميع النتيجة ، استخدمنا متوسط قيمة مائة قراءة.

```
// Makes 100 accelerometer readings and sends the average for smoother result
  angleSum = angleSum + angleInt;
  count++;
```

```
  if (count >= 100) {

    angleInt = angleSum / 100;

    angleSum = 0;

    count = 0;

    // Some more smoothing of acceleromter reading - sends the new angle only
if it differes from the previous one by +-2

    if (angleInt > lastAngle + 2 || angleInt < lastAngle - 2) {

      Serial.println(angleInt);

      String angleString = String(angleInt);

      //sends the angle value with start marker "s" and end marker "e"

      HC12.print("s" + angleString + "e");

      delay(10);

      lastAngle = angleInt;

      angleSum = 0;

      count = 0;

    }

  }
```

لاحظ هنا أنه عند إرسال الزاوية إلى الوحدة النمطية HC-12 ، أرسل أيضًا الحرف "s" في المقدمة ، والحرف "e" بعده ، مما سيساعدني عند تلقي البيانات في Arduino الثاني.

في Arduino الثاني ، ننتظر حتى تأتي علامة البدء "s" ، ثم نقرأ قيمة الزاوية حتى تصل علامة النهاية "e". وبهذه الطريقة ، نحن على يقين من أننا سنحصل فقط على قيمة الزاوية.

```
// Reads the incoming angle

  while (HC12.available()) {             // If HC-12 has data

    incomingByte = HC12.read();          // Store each icoming byte from HC-12

    delay(5);

    // Reads the data between the start "s" and end marker "e"

    if (start == true) {

      if (incomingByte != 'e') {

        readBuffer += char(incomingByte);    // Add each byte to ReadBuffer string variable

      }

      else {

        start = false;

      }

    }

    // Checks whether the received message statrs with the start marker "s"

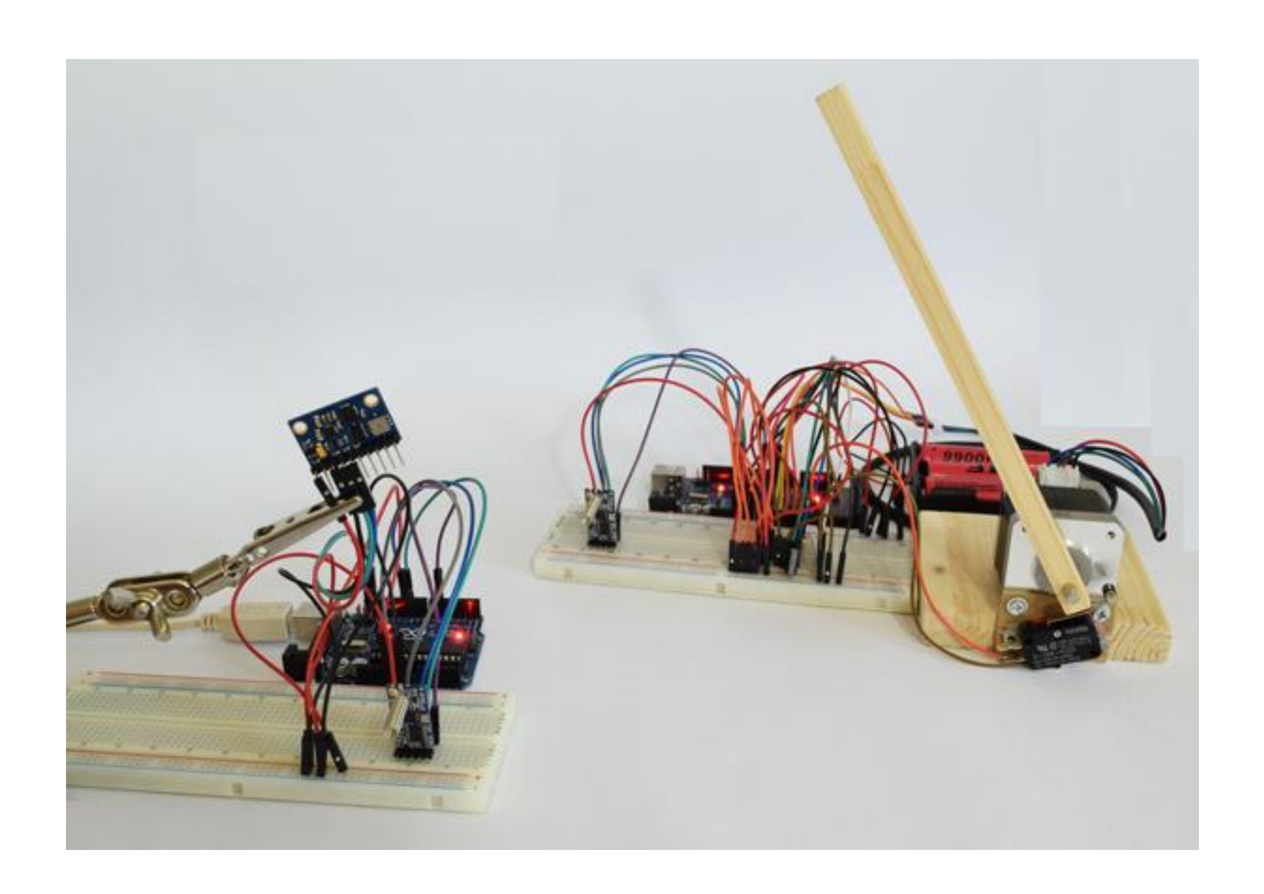
    else if ( incomingByte == 's') {

      start = true; // If true start reading the message

    }

  }
```

ثم نقوم بتحويل القيمة إلى عدد صحيح ، ونقوم بتعيين القيمة من 0 إلى 1600 خطوة والتي تتوافق مع دقة الخطوة السادسة عشرة المحددة في برنامج تشغيل الخطوي A4988 ثم ندير المحرك الخطوي إلى الزاوية الحالية.



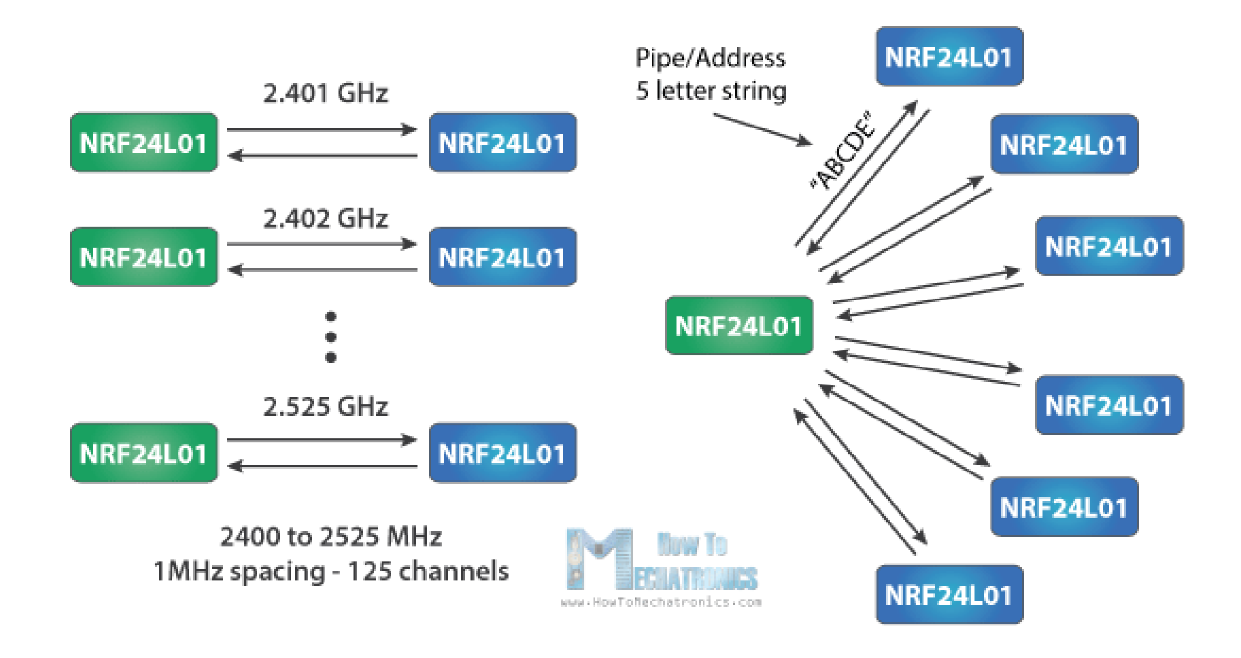
## وحدة الاتصالات اللاسلكية NRF24L01




سنتعلم كيفية إجراء اتصال لاسلكي بين لوحتين من Arduino باستخدام وحدة الإرسال والاستقبال NRF24L01. لشرح الاتصالات اللاسلكية بهذه الوحدة ، سنقدم مثالين ، الأول هو إرسال رسالة "Hello World" بسيطة من لوحة Arduino إلى أخرى ، وفي المثال الثاني ، سيكون لدينا اتصال ثنائي الاتجاه بين لوحات Arduino ، حيث تستخدم عصا التحكم في Arduino الأولى ، سنتحكم في محرك سيرفو في Arduino الثاني ، والعكس بالعكس ، باستخدام زر الضغط في Arduino الثاني ، سنتحكم في LED في Arduino الأول.

دعونا نلقي نظرة فاحصة على وحدة الإرسال والاستقبال NRF24L01. يستخدم النطاق 2.4 جيجا هرتز ويمكن تشغيله بمعدلات باود تتراوح من 250 كيلو بت في الثانية إلى 2 ميغابت في الثانية. إذا تم استخدامه في مساحة مفتوحة وبسرعة منخفضة ، يمكن أن يصل مداها إلى 100 متر.

يمكن للوحدة استخدام 125 قناة مختلفة مما يتيح إمكانية امتلاك شبكة مكونة من 125 مودم يعمل بشكل مستقل في مكان واحد. يمكن أن تحتوي كل قناة على ما يصل إلى 6 عناوين ، أو يمكن لكل وحدة الاتصال بما يصل إلى 6 وحدات أخرى في نفس الوقت.

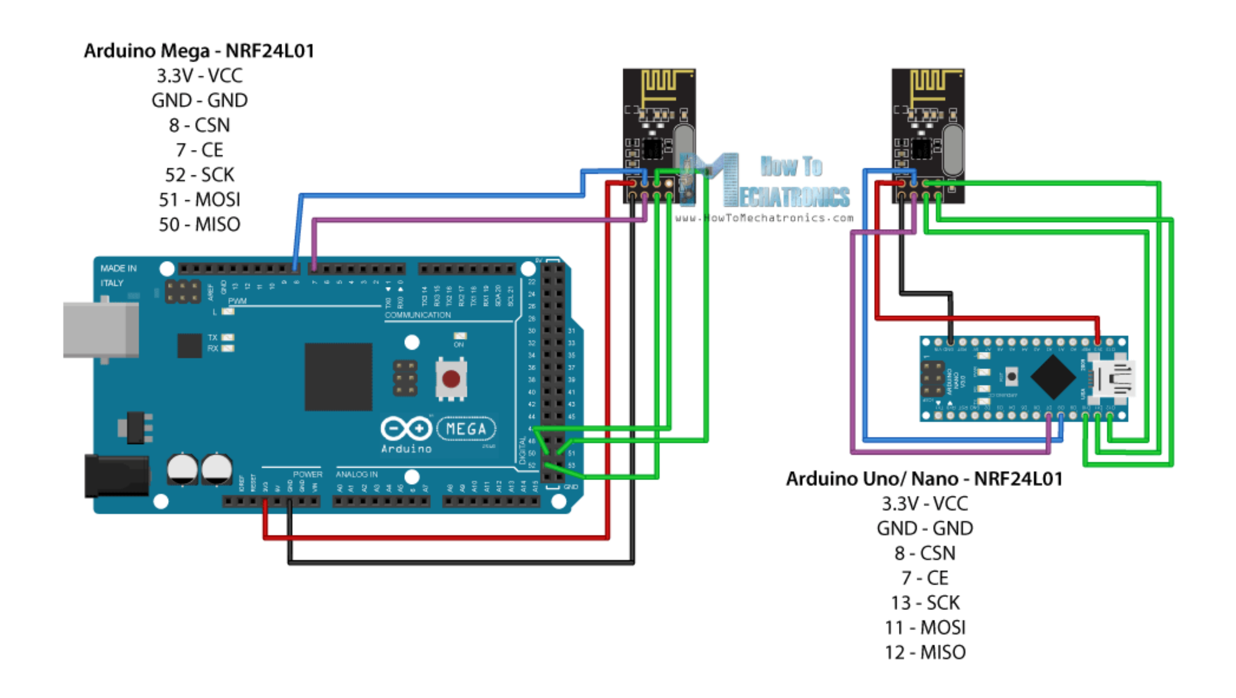



يبلغ استهلاك الطاقة لهذه الوحدة حوالي 12 مللي أمبير أثناء النقل ، وهو أقل من مؤشر LED واحد. يتراوح الجهد التشغيلي للوحدة من 1.9 إلى 3.6 فولت ، ولكن الشيء الجيد هو أن المسامير الأخرى تتسامح مع منطق V5 ، لذلك يمكننا توصيله بسهولة مع Arduino دون استخدام أي محولات مستوى منطقية.



ثلاثة من هذه المسامير مخصصة للاتصال SPI ويجب أن تكون متصلاً بدبابيس SPI الخاصة بـ Arduino ، لكن لاحظ أن كل لوحة Arduino بها دبابيس SPI مختلفة. يمكن توصيل المسامير CSN و CE بأي دبوس رقمي من لوحة Arduino ويتم استخدامها لإعداد الوحدة في وضع الاستعداد أو الوضع النشط standby or active mode ، وكذلك للتبديل بين وضع الإرسال أو وضع الأوامر. آخر دبوس هو دبوس المقاطعة interrupt pin الذي لا يجب استخدامه.

لذلك بمجرد توصيل وحدات NRF24L01 بلوحات Arduino ، فإننا على استعداد لعمل الكودات لكل من جهاز الإرسال والمستقبل.




نحتاج أولاً إلى تنزيل وتثبيت مكتبة RF24 مما يجعل البرمجة أقل صعوبة. فيما يلي الكودات للاتصال اللاسلكي وفيما يلي وصف لهما.

كود الارسال :

```
#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>


RF24 radio(7, 8); // CE, CSN


const byte address[6] = "00001";


void setup() {

  radio.begin();

  radio.openWritingPipe(address);

  radio.setPALevel(RF24_PA_MIN);

  radio.stopListening();

}


void loop() {

  const char text[] = "Hello World";

  radio.write(&text, sizeof(text));

  delay(1000);

}
```

كود الاستقبال :

```
#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>


RF24 radio(7, 8); // CE, CSN


const byte address[6] = "00001";


void setup() {

  Serial.begin(9600);

  radio.begin();

  radio.openReadingPipe(0, address);

  radio.setPALevel(RF24_PA_MIN);

  radio.startListening();

}


void loop() {

  if (radio.available()) {

    char text[32] = "";

    radio.read(&text, sizeof(text));

    Serial.println(text);

  }

}
```

شرح الكود :

نحن بحاجة إلى احتواء SPI الأساسية ومكتبات RF24 المثبتة حديثًا وإنشاء كائن RF24. المعاملات هنا هي CSN و CE دبابيس.

```
RF24 radio(7, 8); // CE, CSN
```

بعد ذلك ، نحتاج إلى إنشاء مصفوفة بايت تمثل العنوان ، أو ما يسمى بالأنبوب الذي سيتم من خلاله التواصل بين الوحدتين.

```
const byte address[6] = "00001";
```

يمكننا تغيير قيمة هذا العنوان إلى أي سلسلة string مكونة من 5 أحرف وهذا يمكّنك من اختيار المتلقي الذي سنتحدث معه ، لذلك في حالتنا سيكون لدينا نفس العنوان في كل من المتلقي والمرسل.

في قسم الإعداد ، نحتاج إلى تهيئة كائن الراديو وباستخدام وظيفة radio.openWritingPipe () نقوم بتعيين عنوان جهاز الاستقبال الذي سنرسل إليه البيانات ، سلسلة string الأحرف المكونة من 5 أحرف التي قمنا بتعيينها مسبقًا.

```
radio.openWritingPipe(address);
```

على الجانب الآخر ، عند المتلقي ، باستخدام وظيفة radio.setReadingPipe () نضع نفس العنوان وبهذه الطريقة يمكننا تمكين الاتصال بين الوحدتين.

```
radio.openReadingPipe(0, address);
```

ثم باستخدام وظيفة radio.setPALevel () قمنا بتعيين مستوى مضخم الطاقة ، وفي حالتنا سوف أضبطها على الحد الأدنى لأن وحداتنا قريبة جداً من بعضها البعض.

```
radio.setPALevel(RF24_PA_MIN);
```

لاحظ أنه في حالة استخدام مستوى أعلى ، يوصى باستخدام المكثفات الالتفافية bypass capacitors عبر GND و 3.3 V من الوحدات بحيث يكون لديها جهد أكثر استقرارًا أثناء التشغيل.

بعد ذلك ، لدينا الامر radio.stopListening () التي تحدد الوحدة كوحدة إرسال ، وعلى الجانب الآخر ، لدينا الامر radio.startListening () التي تحدد الوحدة كمستقبل.

```
// at the Transmitter

radio.stopListening();
```

-

```
// at the Receiver

radio.startListening();
```

في قسم الحلقة "الدوران" ، في جهاز الإرسال ، نقوم بإنشاء مجموعة من الأحرف التي نخصص لها الرسالة "Hello World". باستخدام وظيفة radio.write () سنرسل هذه الرسالة إلى المتلقي. الوسيطة argument الأولى هنا هي المتغير الذي نريد إرساله.

```
void loop() {

 const char text[] = "Hello World";

 radio.write(&text, sizeof(text));

 delay(1000);

}
```

باستخدام "&" قبل اسم المتغير ، قمنا بالفعل بتعيين إشارة إلى المتغير الذي يخزن البيانات التي نريد إرسالها وباستخدام الوسيطة argument الثانية ، نقوم بتعيين عدد البايتات التي نأخذها من هذا المتغير. في هذه الحالة ، تحصل الدالة sizeof () على جميع بايتات السلاسل "text". في نهاية البرنامج ، سنضيف تأخيرًا لمدة ثانية واحدة.

على الجانب الآخر ، في المستقبل ، في قسم الحلقة/الدوران باستخدام وظيفة radio.available () نتحقق مما إذا كانت هناك بيانات يتم استلامها. إذا كان هذا صحيحًا ، فسنقوم أولاً بإنشاء مجموعة/مصفوفة مكونة من 32 عنصرًا ، تسمى "text" ، والتي سنقوم بحفظ البيانات الواردة بها.

```
void loop() {

  if (radio.available()) {

    char text[32] = "";

    radio.read(&text, sizeof(text));

    Serial.println(text);

  }

}
```

باستخدام وظيفة radion.read () نقرأ البيانات ونخزنها في المتغير "text". في النهاية نحن فقط طباعة النص على الشاشة التسلسلية. لذلك بمجرد تحميل البرنامجين ، يمكننا تشغيل الشاشة التسلسلية في جهاز الاستقبال وسوف نلاحظ ظهور رسالة "Hello World" في كل ثانية.

**اتصالات اردوينو ثنائية الاتجاه باستخدام NRF24L01**

لنرى المثال الثاني ، اتصال لاسلكي ثنائي الاتجاه بين لوحتي Arduino. إليكم مخططات الدوائر في الصفحة التالية , علما اننا نحتاج الى القطع التالية :

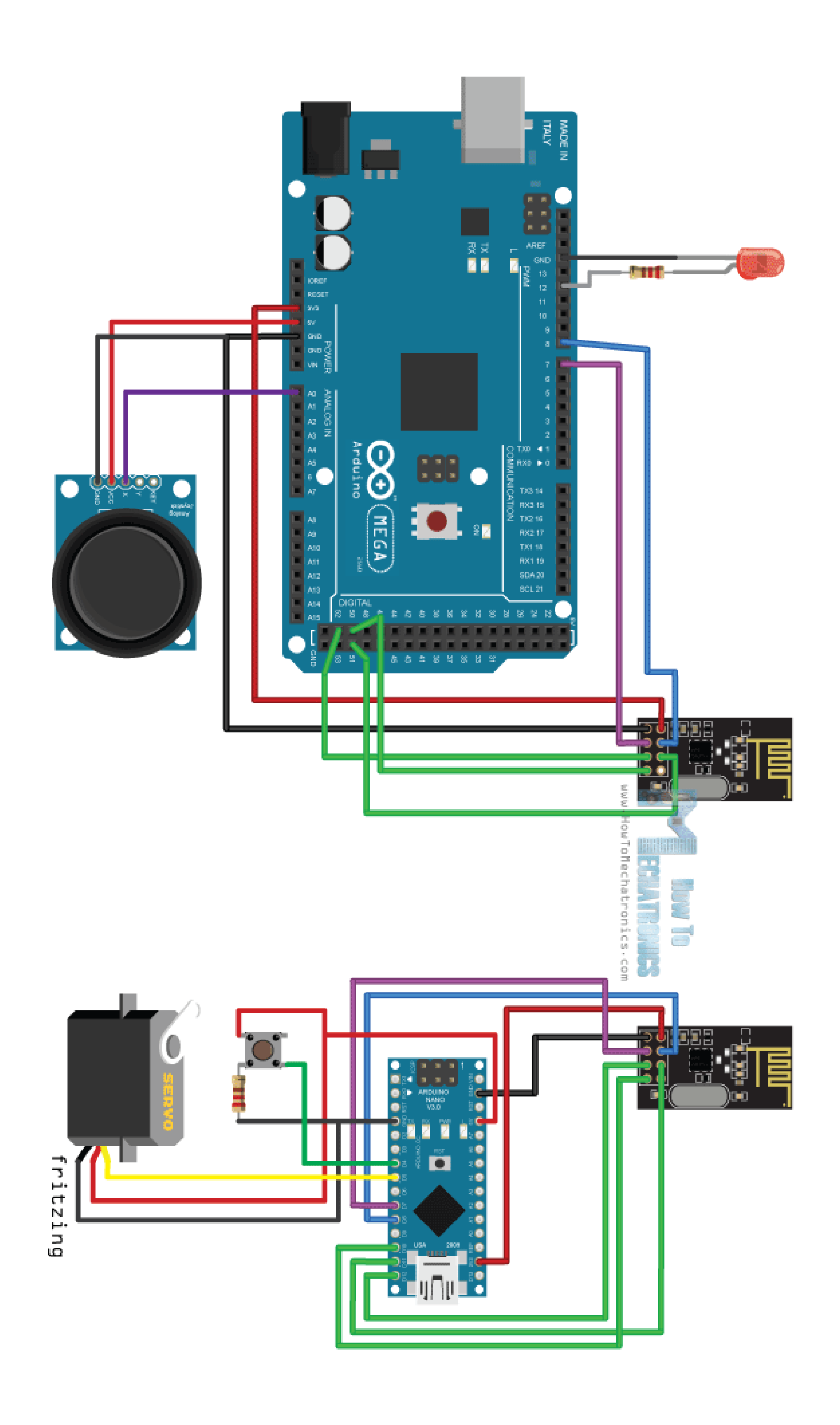
NRF24L01 Transceiver Module

Joystick Module

Servo Motor

LED

Pushbutton



فيما يلي الكودان وأدناه وصفهم.

كود الارسال :

```
#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>


#define led 12


RF24 radio(7, 8); // CE, CSN

const byte addresses[][6] = {"00001", "00002"};

boolean buttonState = 0;


void setup() {

  pinMode(12, OUTPUT);

  radio.begin();

  radio.openWritingPipe(addresses[1]); // 00002

  radio.openReadingPipe(1, addresses[0]); // 00001

  radio.setPALevel(RF24_PA_MIN);

}


void loop() {

  delay(5);


  radio.stopListening();

  int potValue = analogRead(A0);

  int angleValue = map(potValue, 0, 1023, 0, 180);
```

```
  radio.write(&angleValue, sizeof(angleValue));


  delay(5);

  radio.startListening();

  while (!radio.available());

  radio.read(&buttonState, sizeof(buttonState));

  if (buttonState == HIGH) {

    digitalWrite(led, HIGH);

  }

  else {

    digitalWrite(led, LOW);

  }

}
```

كود الاستقبال :

```
#include <SPI.h>

#include <nRF24L01.h>

#include <RF24.h>

#include <Servo.h>


#define button 4


RF24 radio(7, 8); // CE, CSN

const byte addresses[][6] = {"00001", "00002"};

Servo myServo;

boolean buttonState = 0;
```

```
void setup() {
  pinMode(button, INPUT);
  myServo.attach(5);
  radio.begin();
  radio.openWritingPipe(addresses[0]); // 00001
  radio.openReadingPipe(1, addresses[1]); // 00002
  radio.setPALevel(RF24_PA_MIN);
}

void loop() {
  delay(5);
  radio.startListening();
  if ( radio.available()) {
    while (radio.available()) {
      int angleV = 0;
      radio.read(&angleV, sizeof(angleV));
      myServo.write(angleV);
    }
    delay(5);
    radio.stopListening();
    buttonState = digitalRead(button);
    radio.write(&buttonState, sizeof(buttonState));
  }
}
```

الشيء المختلف هنا في المثال السابق هو أننا بحاجة إلى إنشاء أنبوبين أو عناوين للاتصال ثنائي الاتجاه bi-directional communication كالتالي :

```
const byte addresses[][6] = {"00001", "00002"};
```

في قسم الإعداد ، نحتاج إلى تحديد كل من الأنابيب ، ولاحظ أن عنوان الكتابة في Arduino الأول يجب أن يكون عنوان القراءة في Arduino الثاني ، والعكس صحيح ، يجب أن يكون عنوان القراءة في Arduino الأول هو عنوان الكتابة في اردوينو الثاني.

```
// at the Transmitter

radio.openWritingPipe(addresses[1]); // 00001

radio.openReadingPipe(1, addresses[0]); // 00002
```

-

```
// at the Receiver

radio.openWritingPipe(addresses[0]); // 00002

radio.openReadingPipe(1, addresses[1]); // 00001
```

في قسم الحلقة/الدوران باستخدام الامر radio.stopListening () ، قمنا بتعيين أول Arduino باعتباره جهاز إرسال ، وقراءة وتعيين قيمة عصا التحكم من 0 إلى 180 ، وباستخدام وظيفة radio.write () نرسل البيانات إلى المتلقي.

```
radio.stopListening();

int potValue = analogRead(A0);

int angleValue = map(potValue, 0, 1023, 0, 180);

radio.write(&angleValue, sizeof(angleValue));
```

على الجانب الآخر ، باستخدام وظيفة radio.startListening () قمنا بتعيين Arduino الثاني كمستقبل ونتحقق مما إذا كانت هناك بيانات متاحة. في حين أن هناك

بيانات متاحة ، سنقوم بقراءتها وحفظها في متغير "angleV" ثم استخدام هذه القيمة لتدوير محرك السيرفو.

```
radio.startListening();

  if ( radio.available()) {

    while (radio.available()) {

      int angleV = 0;

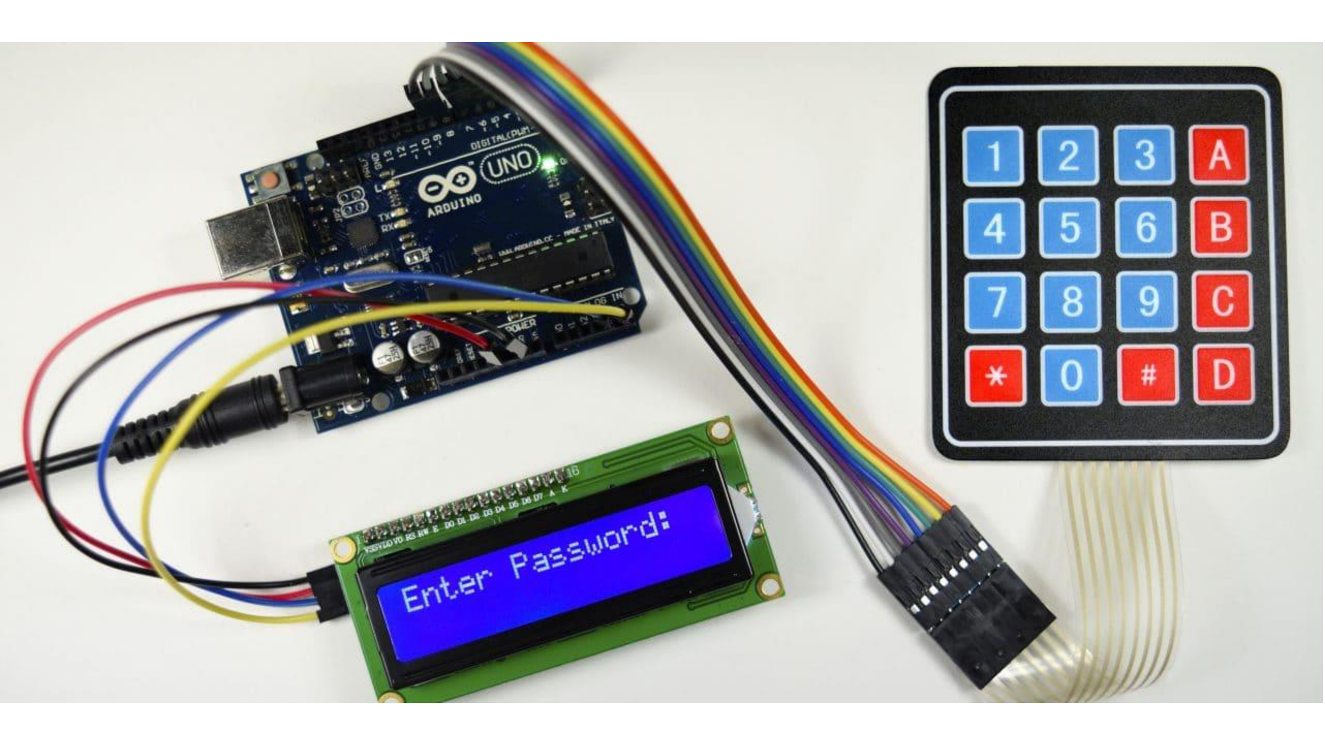
      radio.read(&angleV, sizeof(angleV));

      myServo.write(angleV);

    }
```

بعد ذلك ، في جهاز الإرسال ، قمنا بتعيين أول Arduino كمستقبل ومع وجود حلقة "while" فارغة ، ننتظر وصول بيانات Arduino الثانية ، وهذه هي بيانات حالة زر الضغط سواء تم الضغط عليه أم لا. إذا تم الضغط على الزر ، فسيضيء مؤشر LED. لذلك تتكرر هذه العملية باستمرار وتقوم كل من لوحات Arduino بإرسال واستقبال البيانات باستمرار.

# لوحة المفاتيح



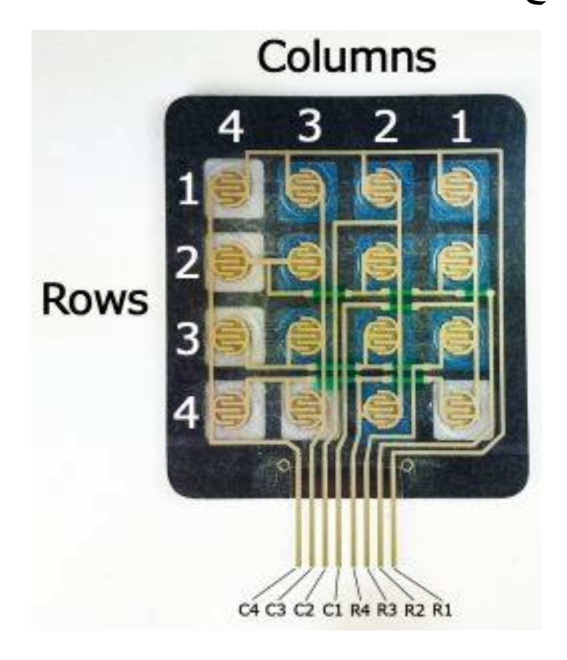

تعد لوحات المفاتيح طريقة رائعة تسمح للمستخدمين بالتفاعل مع مشروعك. يمكنك استخدامها للتنقل بين القوائم وإدخال كلمات المرور والتحكم في الألعاب والروبوتات. سوف نتعلم في هذا الدرس كيفية إعداد لوحة المفاتيح على Arduino. سأشرح أولاً كيف يكتشف Arduino المفاتيح الرئيسية ، ثم سأوضح لك كيفية العثور على pinout لأي لوحة مفاتيح. وكمثال بسيط ، سأوضح لك كيفية طباعة مكابس المفاتيح على الشاشة التسلسلية او شاشة LCD. أخيرًا ، سأوضح لك كيفية تنشيط مرحل V5 عند إدخال كلمة المرور بشكل صحيح.

سأستخدم لوحة مفاتيح بمصفوفة 4*4 ، ولكن هناك أيضًا لوحات المفاتيح التي تعمل بالمصفوفة 3*4. أحب لوحات المفاتيح ذات النمط الغشائي لأنها رقيقة ولديها أيضًا دعامة لاصقة بحيث يمكنك لصقها على معظم الأسطح المستوية. يمكنك أيضًا الحصول على لوحات المفاتيح ذات النمط الهاتفي التي تحتوي على أزرار أكثر سمكًا إذا كنت تحب هذا النمط بشكل أفضل. حتى لوحات المفاتيح التي تم إستخراجها من الهواتف القديمة ستعمل مع Arduino.

## كيف يعملKEYPADS

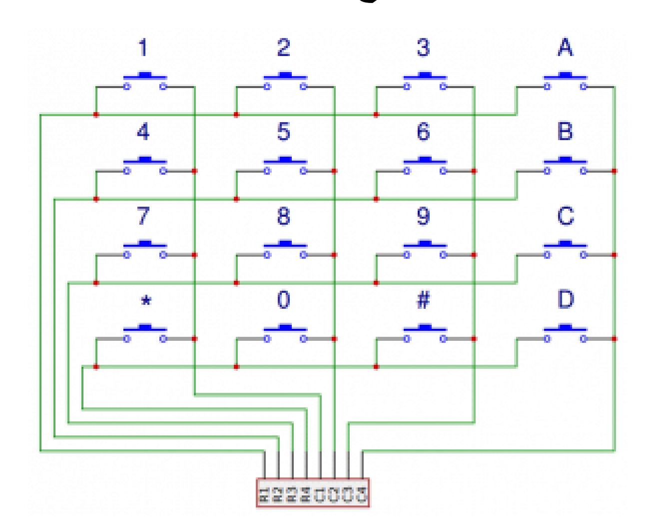



تحت كل مفتاح يوجد غشاء. يتم توصيل كل مفتاح في صف بالمفاتيح الأخرى في الصف بواسطة تتبع موصل أسفل اللوحة. يتم توصيل كل مفتاح في عمود بنفس الطريقة - يتم توصيل جانب واحد من المفتاح بجميع المفاتيح الأخرى في هذا العمود من خلال تتبع موصل. يتم جلب كل صف وعمود إلى دبوس-طرف واحد ، يوجد مجموعه 8 دبابيس على لوحة مفاتيح 4*4



يؤدي الضغط على الزر إلى إغلاق الملامسات بين أحد الأعمدة وتتبعه الصف ، مما يسمح بالتوصيل بين دبوس العمود ودبوس الصف.

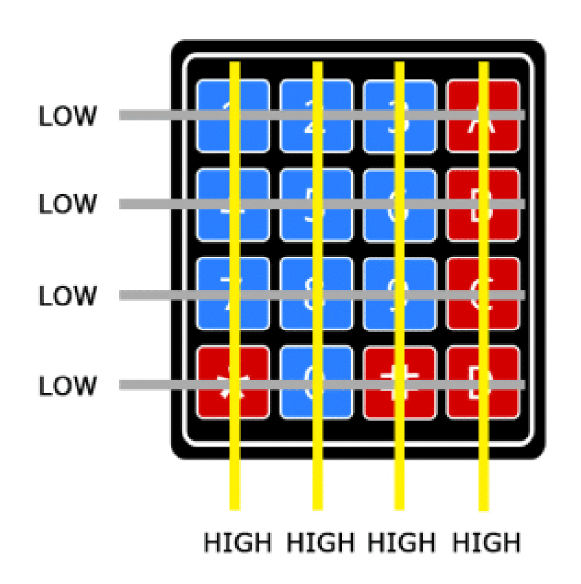
العرض التخطيطي الخاص بلوحة مفاتيح 4*4 كيفية اتصال الصفوف والأعمدة:



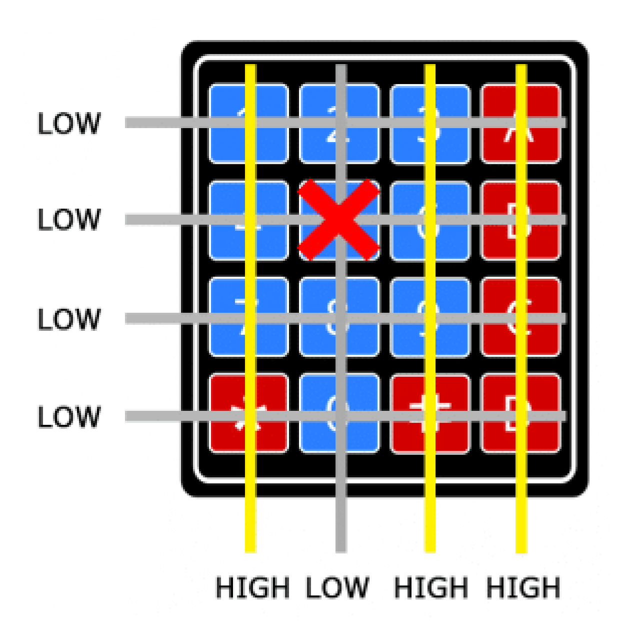

يقوم Arduino بالكشف عن أي زر يتم ضغطه عن طريق الكشف عن الصف ورأس العمود المتصلان بالزر.

وهذا يحدث في أربع خطوات:

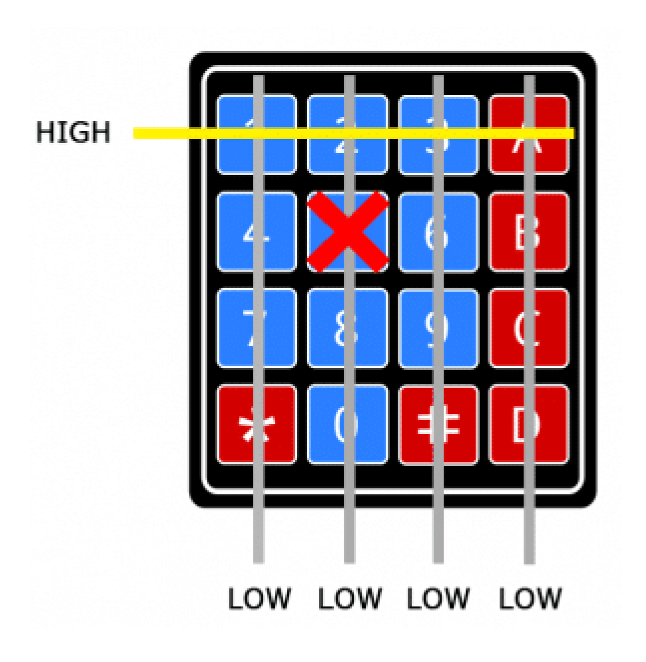
1 – عندما لا يتم الضغط على أي أزرار ، يتم ترميز جميع دبابيس العمود HIGH ، ويتم الاحتفاظ بجميع دبابيس الصفLOW



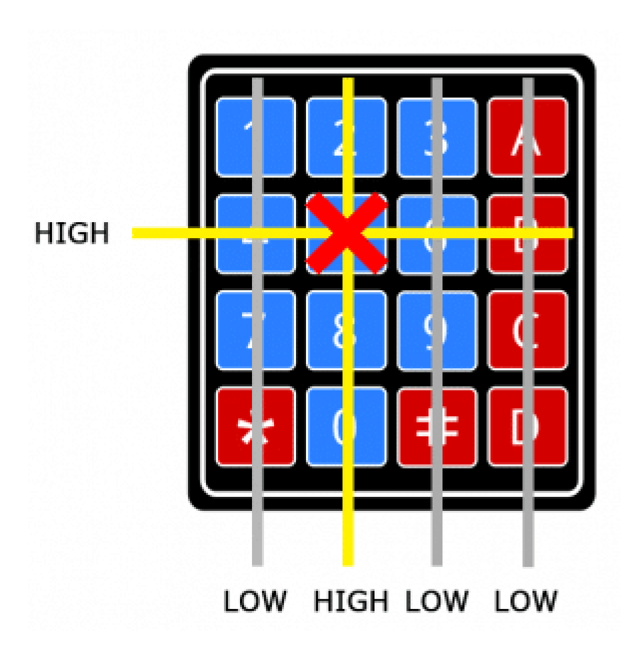
2 – عندما يتم الضغط على زر يتحول العامود الى LOW




3 - يعرف Arduino الآن أي عمود موجود في الزر ، لذلك الآن يحتاج فقط إلى العثور على الصف الموجود في الزر يقوم بذلك عن طريق تبديل كل واحد من دبابيس الصف HIGH ، وفي نفس الوقت قراءة جميع دبابيس الأعمدة للكشف عن أي دبوس العمود يعود إلى HIGH:



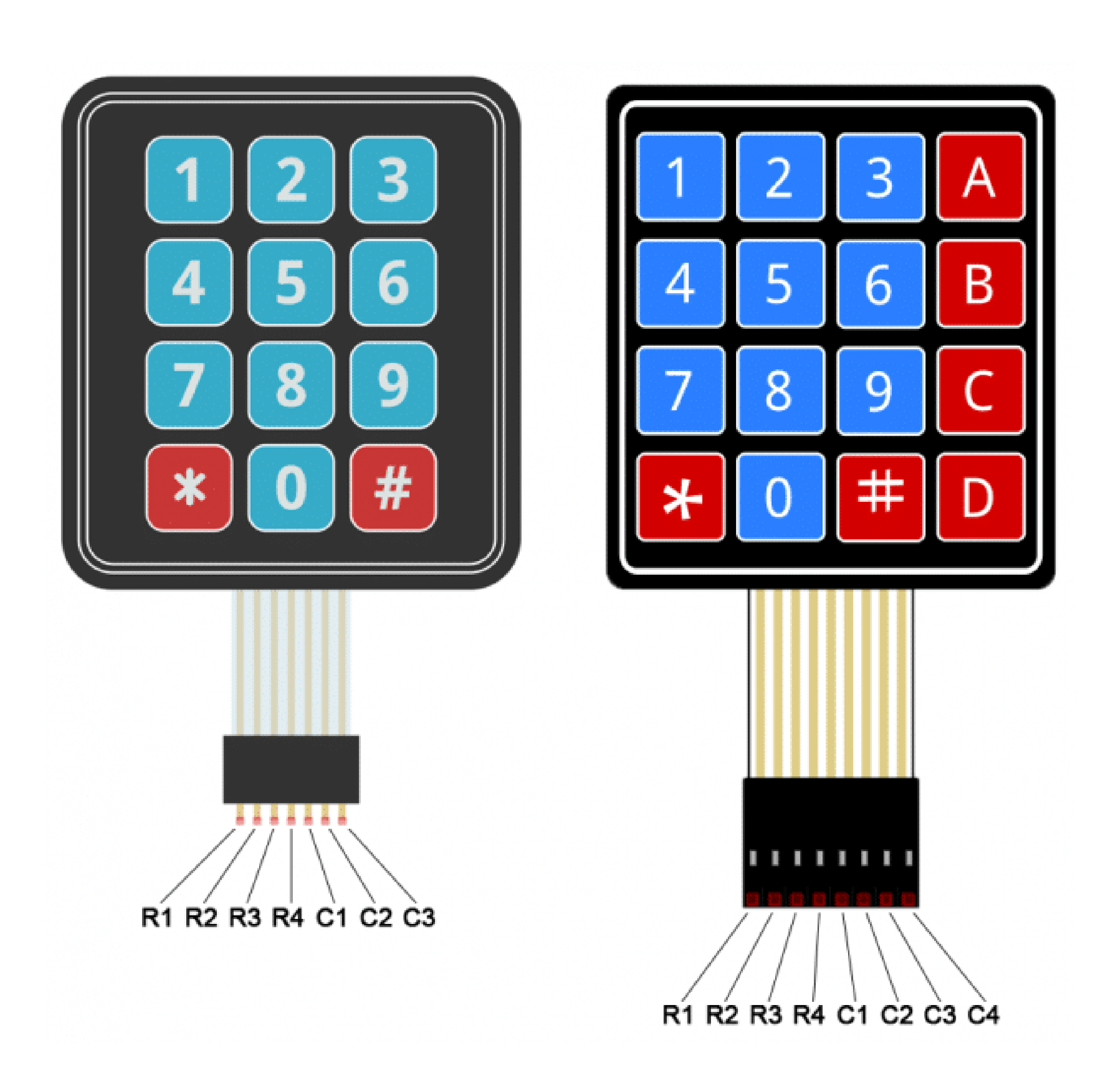
4 - عندما يصبح دبوس العامود HIGH مرة أخرى ، يجد Arduino دبوس الصف المتصل بالزر:



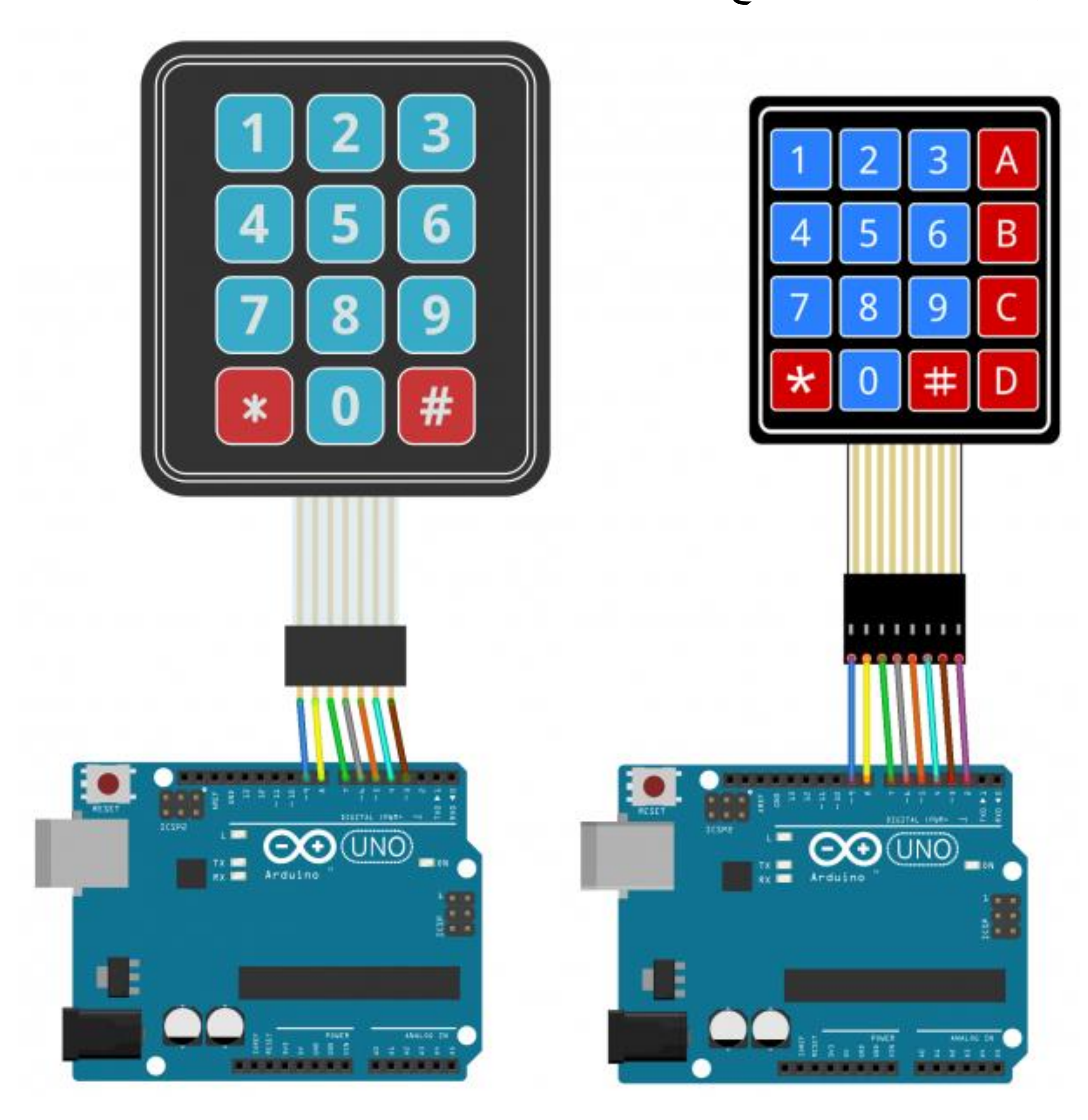

من المخطط أعلاه ، يمكنك أن ترى أن الجمع بين الصف 2 والعمود 2 يمكن أن يعني الضغط فقط على زر الرقم 5.

## ربط ال KEYPAD مع ARDUINO

سيبدو تخطيط الدبابيس لمعظم لوحات المفاتيح الغشائية بالرموز كما يلي:

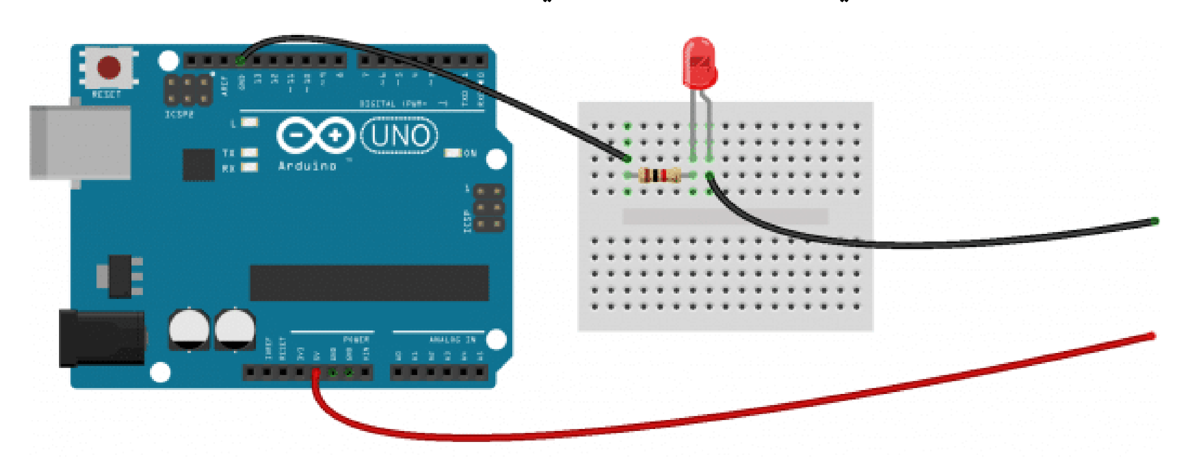


اتبع الرسوم البيانية أدناه لتوصيل لوحة المفاتيح إلى Arduino Uno ، اعتمادًا على ما إذا كان لديك لوحة مفاتيح 4*4 او 4*3



كيف تجدPINOUT المخرج من KEYPAD الخاص بك!

إذا كان تخطيط دبوس لوحة المفاتيح لا يتطابق مع المخططات المذكورة أعلاه ، فيمكنك فحص الأطراف لتحديدها. ستحتاج إلى إنشاء دائرة اختبار عن طريق توصيل LED ومقاومة بمصدر خارجي 5 Vعلى النحو التالي :




أولاً ، تعرّف على دبابيس لوحة المفاتيح المتصلة بصفوف الزر. أدخل السلك الأرضي (الأسود) في الدبوس الأول على اليسار. اضغط على أي زر في الصف 1 وثبت الضغط.

الآن أدخل السلك الموجب (الأحمر) في كل واحد من المسامير الأخرى. إذا كان مصباح LED يضيء عند أحد المسامير ، فاضغط مع الاستمرار على زر آخر في الصف 1 ، ثم أدخل السلك الموجب في كل مسامير أخرى مرة أخرى.

إذا كان مصباح LED يضيء على دبوس مختلف ، فهذا يعني أن السلك الأرضي يتم إدخاله في مسمار الصف 1.

إذا لم يعمل أي من الأزرار الموجودة في الصف 1 على إضاءة مصباح LED ، فإن السلك الأرضي غير متصل بالصف 1.

الآن انقل السلك الأرضي إلى الطرف التالي ، واضغط على زر في صف مختلف ، وكرر العملية أعلاه حتى تعثر على الدبوس لكل صف.

لمعرفة دبابيس الأعمدة المتصلة ، أدخل السلك الأرضي في الدبوس الذي تعرفه في الصف 1.

الآن اضغط مع الاستمرار على أي من الأزرار الموجودة في هذا الصف.

الآن أدخل السلك الإيجابي في كل واحد من الدبابيس المتبقية. الدبوس الذي يضيء مصباح LED هو دبوس متصل بعمود ذلك الزر.

الآن اضغط على زر آخر في نفس الصف ، وأدخل السلك الموجب في كل واحد من المسامير الأخرى.

كرر هذه العملية لكل واحد من الأعمدة الأخرى حتى يتم تعيين كل واحد منها.

للحصول على عرض توضيحي أساسي حول كيفية إعداد لوحة المفاتيح ، سأوضح لك كيفية طباعة كل مفتاح من مفاتيح الضغط على الشاشة التسلسلية.

سنستخدم مكتبة لوحة المفاتيح بواسطة Mark Stanley و Alexander Brevig. تعتني هذه المكتبة بإعداد الدبابيس وتوقع الأعمدة والصفوف المختلفة. لتثبيت مكتبة لوحة المفاتيح ، انتقل إلى Sketch> Include Library> Manage Libraries وابحث عن "keypad". انقر فوق المكتبة ، ثم انقر فوق install .

http://playground.arduino.cc/Code/Keypad

بمجرد تثبيت مكتبة لوحة المفاتيح ، يمكنك تحميل هذا الكود إلى Arduino إذا كنت تستخدم لوحة مفاتيح 4*4

```
#include <Keypad.h>



const byte ROWS = 4;

const byte COLS = 4;



char hexaKeys[ROWS][COLS] = {

  {'1', '2', '3', 'A'},

  {'4', '5', '6', 'B'},

  {'7', '8', '9', 'C'},
```

```
  {'*', '0', '#', 'D'}

};



byte rowPins[ROWS] = {9, 8, 7, 6};

byte colPins[COLS] = {5, 4, 3, 2};



Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS,
COLS);



void setup(){

  Serial.begin(9600);

}



void loop(){

  char customKey = customKeypad.getKey();



  if (customKey){

    Serial.println(customKey);

  }

}
```

إذا كنت تستخدم لوحة مفاتيح 4*3، فيمكنك استخدام هذا الرمز:

```
#include <Keypad.h>



const byte ROWS = 4;

const byte COLS = 3;
```

```
char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};


byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3};


Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);


void setup(){
  Serial.begin(9600);
}


void loop(){
  char customKey = customKeypad.getKey();

  if (customKey){
    Serial.println(customKey);
  }
}
```

تعيين الأسطر 3 و 4 في البرمجية أعلاه عدد الصفوف والأعمدة على لوحة المفاتيح.

تحدد السطور 6-11 الأحرف التي يتم طباعتها عند الضغط على زر معين على لوحة المفاتيح. يتم وضع الأحرف تمامًا كما تظهر على لوحة المفاتيح. إذا كانت لوحة المفاتيح الخاصة بك تحتوي على تخطيط مختلف ، فيمكنك تحديد الأحرف التي يتم طباعتها عند الضغط على زر. على سبيل المثال ، لنفترض أن لوحة المفاتيح تحتوي على عمود من الأحرف على اليسار بدلاً من اليمين. عليك فقط تغييره إلى هذا:

```
  char hexaKeys[ROWS][COLS] = {

{'A', '1', '2', '3'},

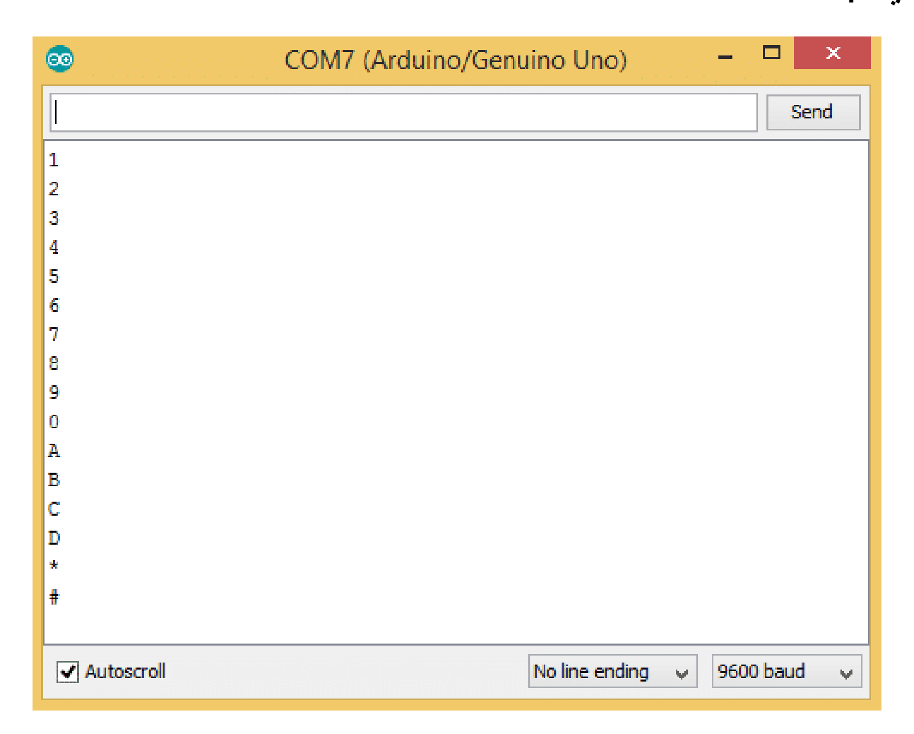
{'B', '4', '5', '6'},

{'C', '7', '8', '9'},

{'D', '*', '0', '#'}

};
```

بعد تحميل الرمز ، افتح الشاشة التسلسلية. عند الضغط على أحد المفاتيح ، ستتم طباعة القيمة:



## استخدام LCD مع KEYPAD

لنرى الآن كيفية طباعة مفاتيح الضغط على شاشة LCD. تستخدم لوحات المفاتيح X4 84 دبابيس ولوحات المفاتيح X43 تستخدام 7 دبابيس. يستهلك هذا الكثير من الدبابيس ، لذا سأستخدم شاشة LCD مزودة بتقنية I2C لأنها تحتاج فقط إلى 4 أسلاك للاتصال بـ Arduino.

### تثبيت مكتبة LIQUIDCRYSTAL_I2C

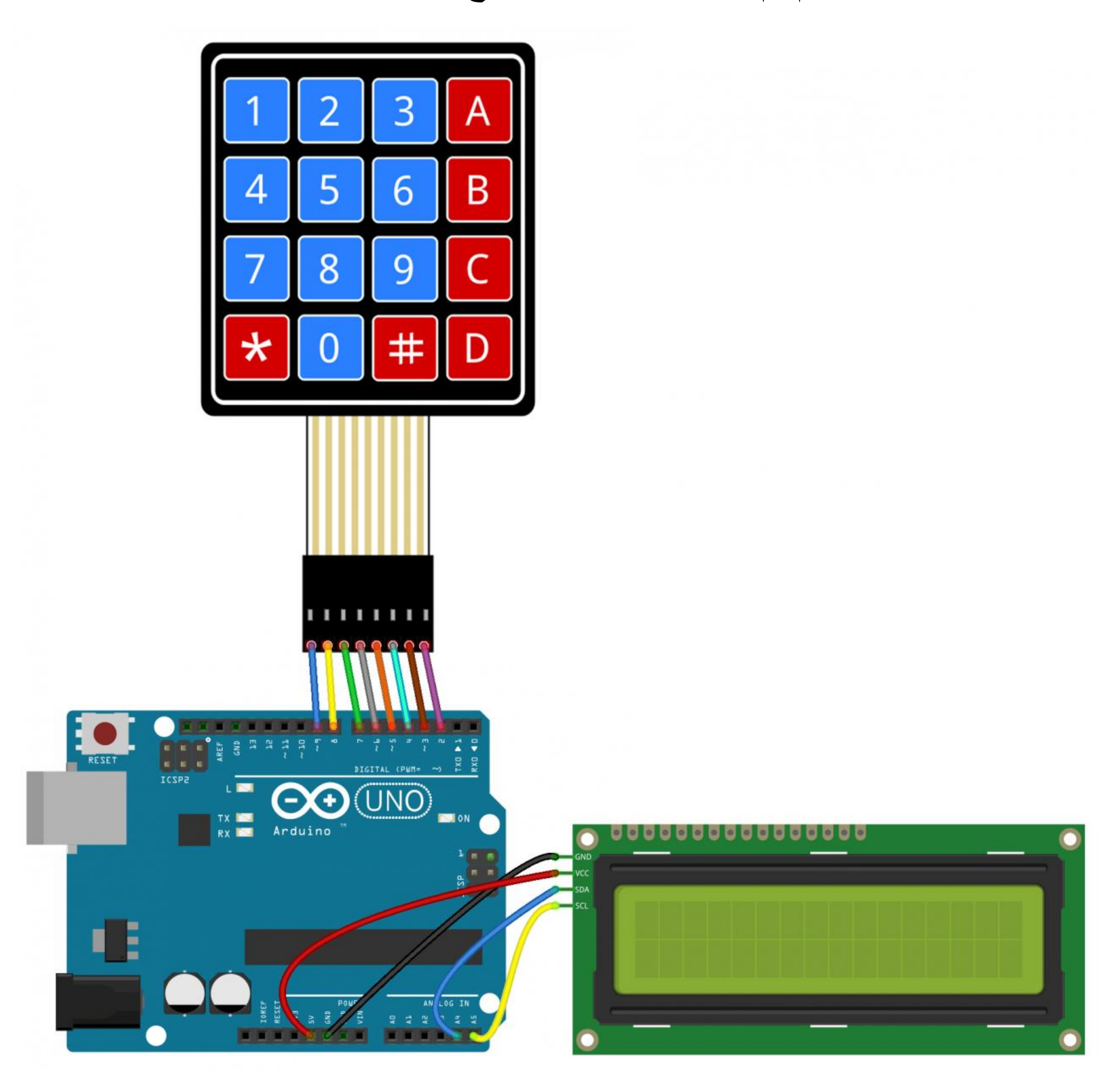
https://github.com/johnrickman/LiquidCrystal_I2C

هناك حاجة إلى مكتبة الأسلاك لإضافة دعم للاتصال I2C. يأتي مزودًا بخاصية Arduino IDE ، لذلك لا تحتاج إلى تثبيته. ولكن إذا لم يكن مثبتًا على نظامك لسبب ما ، فانتقل إلى Sketch> Include Library> Manage Libraries وابحث عن "wire" لتثبيته.

بمجرد تثبيت المكتبات ، قم بتوصيل المسامير الأرضية و Vcc لشاشة LCD إلى Arduino، ثم قم بتوصيل مسامير SDA و SCL الخاصة بشاشات LCD وفقًا للجدول أدناه الخاص بلوحات Arduino المختلفة:

| Arduino | SDA Pin | SCL Pin |
|---|---|---|
| Uno | A4 | A5 |
| Nano | A4 | A5 |
| Mini | A4 | A5 |
| 101 | SDA | SCL |
| Zero | SDA | SCL |
| Leonardo | 2 | 3 |
| Micro | 2 | 3 |
| Due | 20 | 21 |
| Mega | 20 | 21 |

ثم قم بتوصيل لوحة المفاتيح إلى اردوينو.




بمجرد توصيل كل شيء ، قم بتحميل هذا الكود علىArduino

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

#include <Keypad.h>
```

```
const byte ROWS = 4;

const byte COLS = 4;


char hexaKeys[ROWS][COLS] = {

  {'1', '2', '3', 'A'},

  {'4', '5', '6', 'B'},

  {'7', '8', '9', 'C'},

  {'*', '0', '#', 'D'}

};


byte rowPins[ROWS] = {9, 8, 7, 6};

byte colPins[COLS] = {5, 4, 3, 2};


Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS,
COLS);


LiquidCrystal_I2C lcd(0x21, 16, 2);


void setup(){

  lcd.backlight();

  lcd.init();

}


void loop(){

  char customKey = customKeypad.getKey();

  if (customKey){

    lcd.clear();
```

```
    lcd.setCursor(0, 0);

    lcd.print(customKey);

  }

}
```

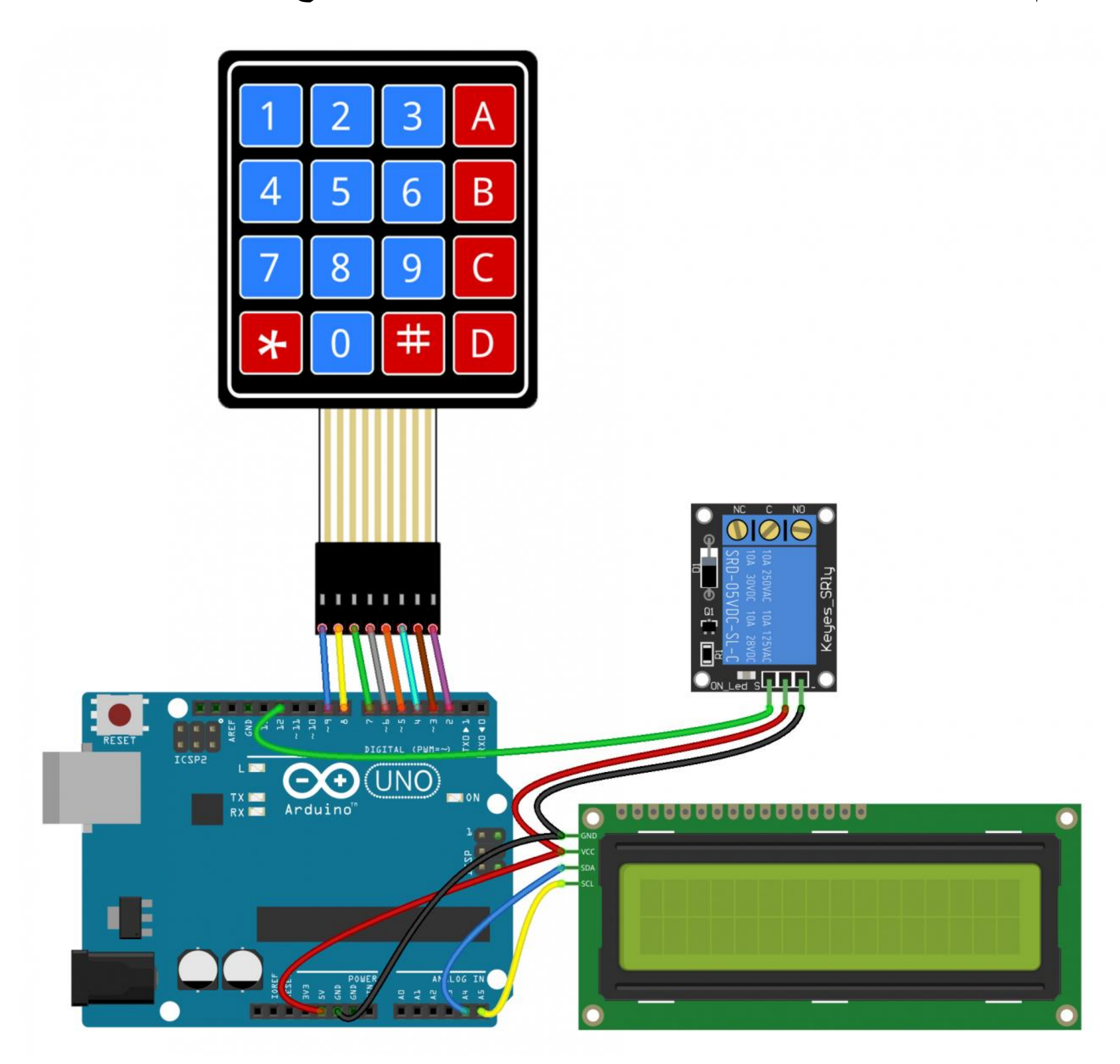
ستحتاج إلى إضافة عنوان I2C لشاشة LCD على السطر 20:

```
LiquidCrystal_I2C lcd(0x21, 16, 2);
```

عنوان I2C لشاشتي LCD هو x210 ، لكن من المحتمل أن يكون عنوانك مختلفًا. يجب توفير عنوان I2C لشاشة LCD في ورقة البيانات ، ولكن إذا لم يكن الأمر كذلك ، فيمكنك العثور عليه عن طريق رسم I2C_Scanner هذا.

استخدام كلمة المرور لتنشيط مرحل

أحد أكثر التطبيقات المفيدة في لوحة المفاتيح هو استخدامه لادخال المفاتيح لتنفذ امر. يمكنك إعداد كلمة مرور وتفعيل Arduino للتتابع إذا كانت كلمة المرور صحيحة. سيقوم بتنشيط مرحل 5 Vعند إدخال كلمة المرور بشكل صحيح فقط:



بمجرد توصيل كل شيء ، قم بتحميل هذا الكود علىArduino

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

#include <Keypad.h>


#define Password_Length 8


int signalPin = 12;


char Data[Password_Length];

char Master[Password_Length] = "123A456";

byte data_count = 0, master_count = 0;

bool Pass_is_good;

char customKey;


const byte ROWS = 4;

const byte COLS = 4;


char hexaKeys[ROWS][COLS] = {

  {'1', '2', '3', 'A'},

  {'4', '5', '6', 'B'},

  {'7', '8', '9', 'C'},

  {'*', '0', '#', 'D'}

};


byte rowPins[ROWS] = {9, 8, 7, 6};
```

```
byte colPins[COLS] = {5, 4, 3, 2};


Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS,
COLS);


LiquidCrystal_I2C lcd(0x21, 16, 2);


void setup(){

  lcd.init();

  lcd.backlight();

  pinMode(signalPin, OUTPUT);

}


void loop(){


  lcd.setCursor(0,0);

  lcd.print("Enter Password:");


  customKey = customKeypad.getKey();

  if (customKey){

    Data[data_count] = customKey;

    lcd.setCursor(data_count,1);

    lcd.print(Data[data_count]);

    data_count++;

    }


  if(data_count == Password_Length-1){
```

```
    lcd.clear();


    if(!strcmp(Data, Master)){
      lcd.print("Correct");
      digitalWrite(signalPin, HIGH);
      delay(5000);
      digitalWrite(signalPin, LOW);
      }
    else{
      lcd.print("Incorrect");
      delay(1000);
      }


    lcd.clear();
    clearData();
  }
}


void clearData(){
  while(data_count !=0){
    Data[data_count--] = 0;
  }
  return;
}
```

يمكنك تغيير كلمة المرور على السطر 10 عن طريق استبدال النص A456123 بكلمة المرور الخاصة بك:

```
char Master[Password_Length] = “123A456”;
```

يجب تعيين طول كلمة المرور على السطر 5:

```
#define Password_Length 8
```

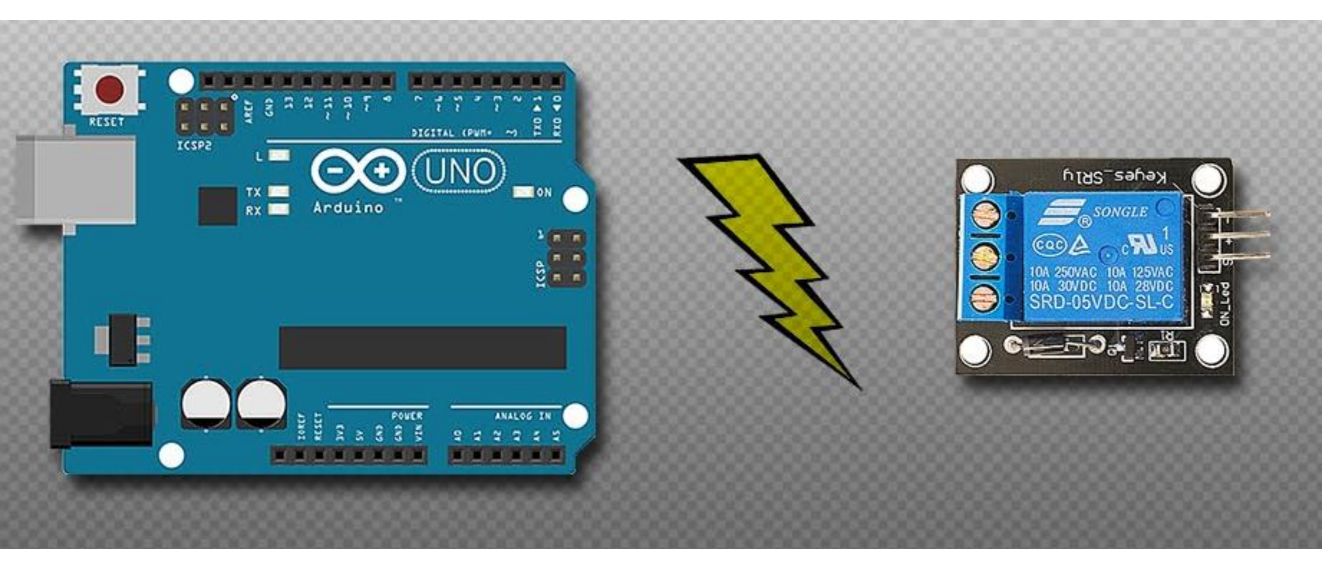
كلمة المرور في المثال أعلاه عبارة عن 7 أحرف فقط ، ولكن طول كلمة المرور أكبر من 7 لأن هناك حرفًا فارغًا مضافًا إلى نهاية string. على سبيل المثال ، إذا كانت كلمة المرور الخاصة بك 5 أحرف ، فستدخل الرقم 6 لطول كلمة المرور.

يتم تحديد دبوس الإخراج الذي يشغل المرحل على السطر 7:

```
int signalPin = 12;
```

حسنا هذا هو بما يتعلق ب لوحة المفاتيح. لاحظ انه ليس من الصعب إعداد لوحة مفاتيح على الإطلاق. أعتقد أنه باستخدام القليل من التجربة والخطأ ، يجب أن تكون قادرًا على تعديل الكودات أعلاه للعمل مع معظم المشاريع التي ترغب في استخدام لوحة مفاتيح لها.

## المرحلات Relays




واحدة من أكثر الأشياء المفيدة التي يمكنك القيام بها باستخدام Arduino هي التحكم بأجهزة الجهد العالي (120-V240) مثل المراوح والأضواء وأجهزة التسخين وغيرها من الأجهزة المنزلية. نظرًا لأن نظام Arduino يعمل عند 5 فولت ، فإنه لا يمكن التحكم في هذه الأجهزة ذات الجهد العالي مباشرةً ، ولكن يمكنك استخدام مرحل 5 فولت لتشغيل جهد 120-240 فولت واستخدام Arduino للتحكم في المرحل ببساطة.

يمكن برمجة Arduino لتشغيل المرحل عند حدوث حدث معين ، على سبيل المثال عندما تكون درجة حرارة الثرمستور أعلى من 30 درجة مئوية أو عندما تنخفض مقاومة المقاومة الضوئية إلى أقل من 400 أوم. يمكن استخدام أي مستشعر تقريبًا لتشغيل المرحل لتشغيله أو إيقاف تشغيله. لا يحتاج حتى ليكون من جهاز استشعار. فيمكن أن تحدث في فترات زمنية محددة ، او يمكن تشغيلها من خلال الضغط على زر او باستخدام الهاتف أو حتى عند تلقي بريد إلكتروني.

في هذا الدرس سوف أستخدم المرحل SRD-05VDC-SL-C 5V Relay لأنه يحظى بشعبية كبيرة بين الهواة و لنبدأ برؤية كيفية عمل تتابع 5 V، ثم سأوضح كيفية إعداده على Arduino وسنقدم لك بعض الكودات لتشغيله.

كيف يعمل مرحل ال V5

يحتوي المرحل SRD-05VDC-SL-C على ثلاث دبابيس جهد عالي ( NC، C ، و NO) التي تتصل بالجهاز الذي تريد التحكم فيه. ويحتوي الطرف الآخر على ثلاثة دبابيس ذات فولتية منخفضة (GND و Vcc وإشارة) والتي تتصل بـ Arduino.




- NC: Normally closed 120-240V terminal
- NO: Normally open 120-240V terminal
- C: Common terminal
- Ground: Connects to the ground pin on the Arduino
- 5V Vcc: Connects the Arduino’s 5V pin
- Signal: Carries the trigger signal from the Arduino that activates the relay

يوجد داخل المرحل مفتاح 120-240 فولت متصل بمغناطيس كهربائي. عندما يتلقى المرحل إشارة HIGH في دبوس الإشارة ، يصبح المغناطيس الكهربائي مشحونًا وينقل جهات الاتصال الخاصة بالمفتاح مفتوحًا أو مغلقًا.

ما هي NORMALLY OPEN وNORMALLY CLOSED .

يحتوي المرحل على نوعين مختلفين من الاتصالات الكهربائية في الداخل - عادة مفتوحة (NO) او عادة مغلقة (NC).يعتمد النوع الذي تستخدمه على ما إذا كنت تريد أن تقوم إشارة 5 Vبتشغيل المفتاح أو إيقاف تشغيل المفتاح. دخل التيار 120-240 فولت يدخل في التتابع في المحطة المشتركة (C) في كلا التكوينين.لاستخدام جهات الاتصال المفتوحة عادة ، استخدم NO terminal لاستخدام جهات الاتصال المغلقة عادة ، استخدم NC terminal .

عادة مفتوحة NORMALLY OPEN

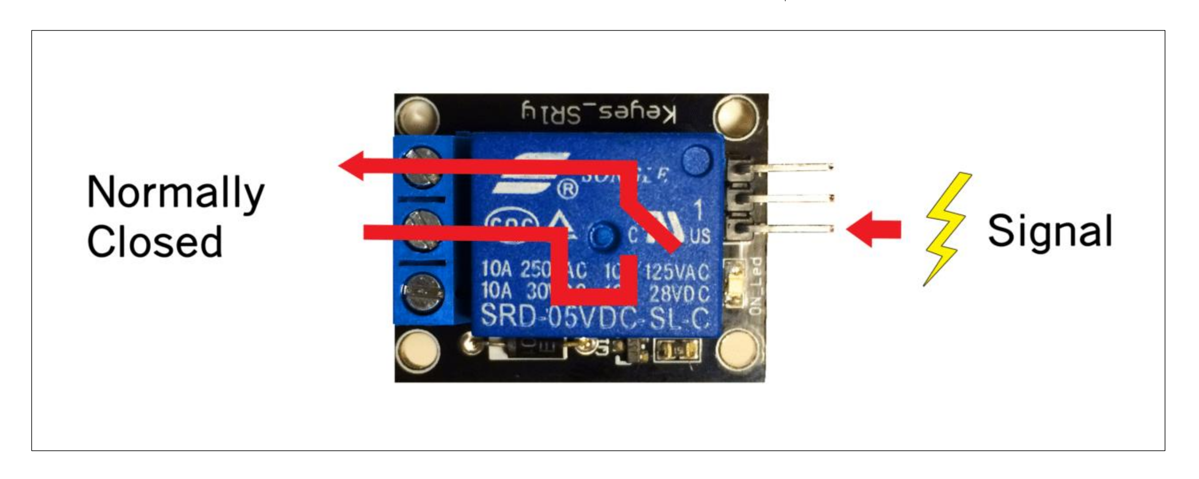
في التكوين المفتوح عادة ، عندما يتلقى المرحل إشارة HIGH ، يغلق المفتاح 120-V240 ويسمح للتيار بالتدفق من الطرف C إلى الطرف NO.

تقوم إشارة LOW بإلغاء تنشيط المرحل وتوقف التيار. لذلك إذا كنت تريد أن تقوم إشارة HIGH بتشغيل وحدة المرحل ، فاستخدم المحطة المفتوحة عادة



## عادة مغلقة NORMALLY CLOSED

في التكوين المغلق عادة ، تفتح إشارة المرحل مفتاح التبديل وتقاطع التيار 120-240 فولت. تغلق إشارة منخفضة المفتاح وتسمح للتيار بالتدفق من الطرف C إلى الطرف NC. لذلك ، إذا كنت تريد أن تقوم إشارة HIGH بإيقاف تشغيل التيار 120-240 فولت ، فاستخدم المحطة المغلقة عادة

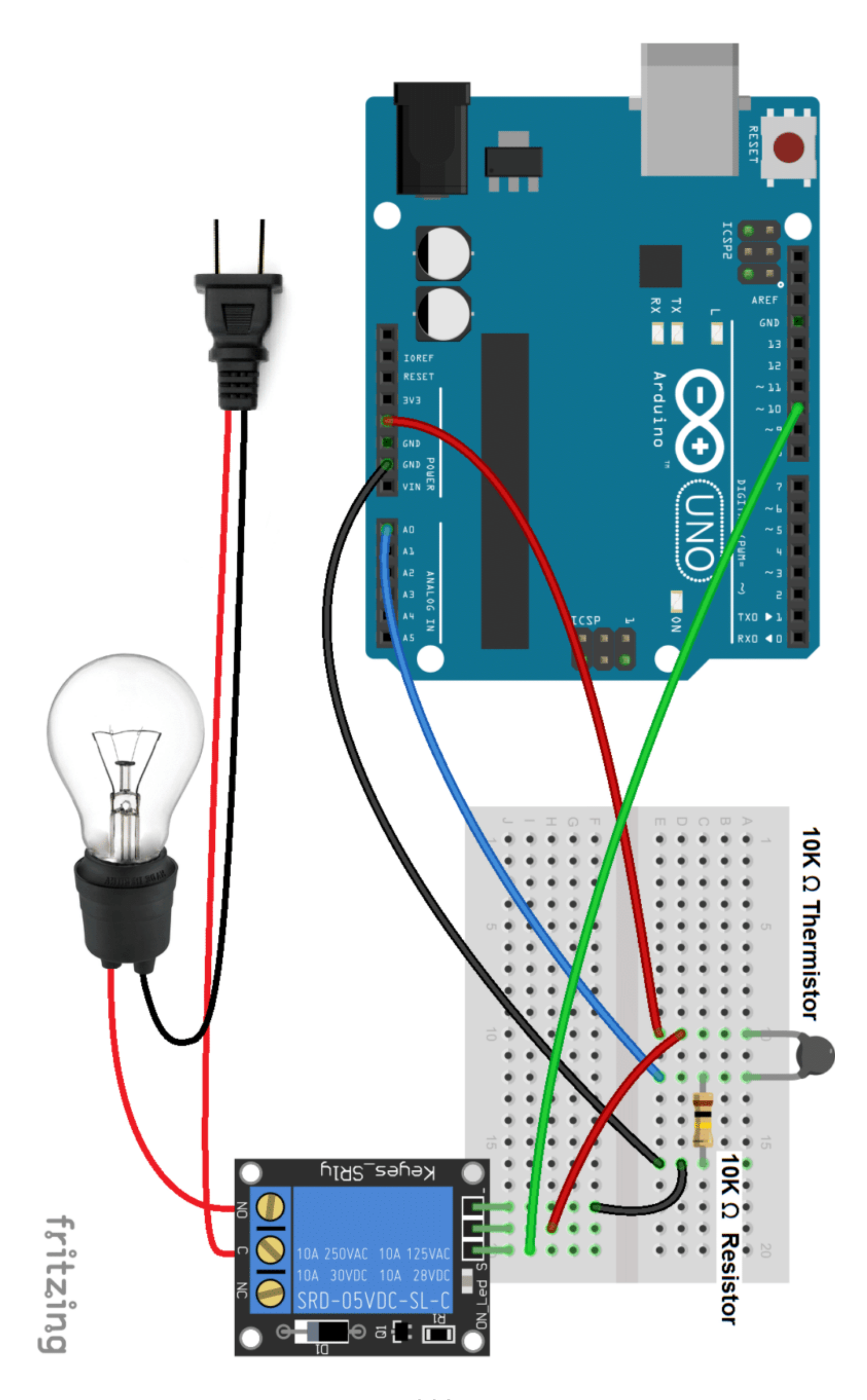



## درجة الحرارة للتحكم في دائرة مرحل V5

لتظهر لك كيفية توصيل المرحل ، دعنا نبني دائرة مرحل يتم التحكم في درجة حرارتها والتي ستقوم بإطفاء مصباح كهربائي عندما تصل درجة حرارة الثرمستور إلى 150 درجة فهرنهايت.

الترمستورات مفيدة حقا مع المرحلات V5. يمكنك استخدامها لإيقاف تشغيل محرك كبير في حالة السخونة أو تشغيل السخان إذا كانت درجة الحرارة شديدة البرودة.

الإعداد بسيطة للغاية ، فقط تأكد من أن وصلات الجهد العالي موصلة بطريقة آمنة:



حدد سلك الطاقة (السلك الأحمر في الرسم أعلاه) في السلك المؤدي إلى المصباح الكهربائي وقم بعمل قطع.

قم بتوصيل الجانب المؤدي إلى المصباح الكهربائي إلى طرفية NO للمرحل ، والجانب المؤدي إلى القابس بالطرف C.

بهذه الطريقة يكون التتابع على الجانب الساخن ، ويتم تبديل التيار قبل وصوله إلى المصباح الكهربائي. من الخطر وضع الترحيل على السلك المحايدneutral ، حيث أنه إذا فشل الجهاز الحالي ، فلا يزال من الخطأ توجيهه عند إيقاف التتابع.

يتم تعيين جزء الثرمستور من الدائرة كمقسم الجهد. يجب أن تكون قيمة المقاوم نفس ترتيب قيمة الثرمستور. على سبيل المثال ، أنا أستخدم الثرمستور K10، لذلك يجب أن يكون المقاوم K Ω10 كذلك. إذا كنت تستخدم الثرمستور K100، استخدم المقاوم .K100

إذا كنت تستخدم الثرمستور K100، فستحتاج إلى تغيير السطر 7 في الكود أدناه

```
Temp = log(100000.0*((1024.0/RawADC-1)));
```

بعد توصيل كل شيء ، حمّل هذا الكود إلىArduino

```
#include <math.h>



int pinOut = 10;



double Thermistor(int RawADC) {

 double Temp;

 Temp = log(10000.0*((1024.0/RawADC-1)));

 Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 *
Temp * Temp ))* Temp );
```

```
 Temp = Temp - 273.15;
 Temp = (Temp * 9.0)/ 5.0 + 32.0;
 return Temp;
}

void setup() {
  Serial.begin(9600);
  pinMode(10, OUTPUT);
}

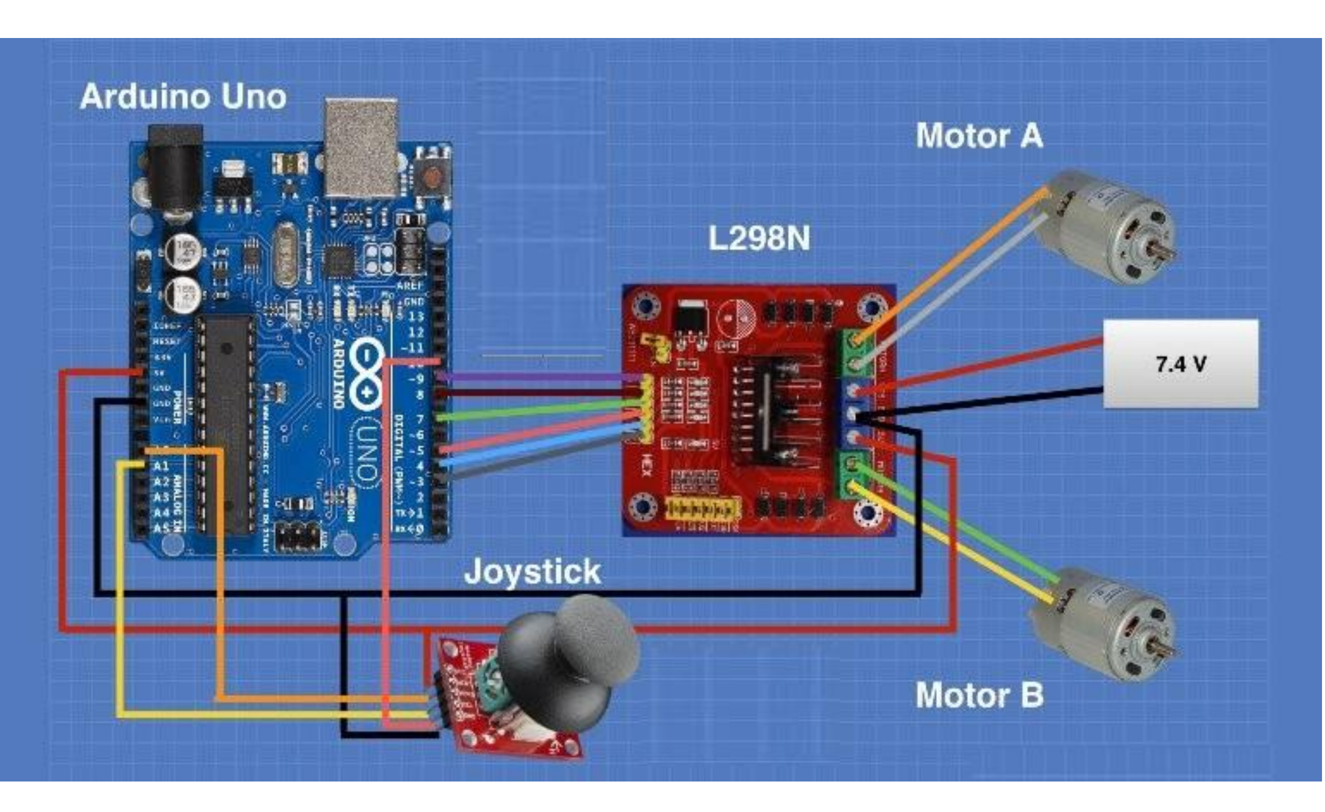
void loop() {
  int val;
  double temp;
  val=analogRead(0);
  temp=Thermistor(val);
  Serial.print("Temperature = ");
  Serial.print(temp);
  Serial.println(" F");
  if (temp >= 150){
    digitalWrite(pinOut, LOW);
  }
  else {
    digitalWrite(pinOut, HIGH);
  }
```

```
  delay(500);

}
```

في هذا المثال ، سيبقى المرحل نشطًا وسيسمح بتدفق التيار من خلال المصباح الكهربائي حتى تصل درجة حرارة الثرمستور إلى 150 درجة فهرنهايت. عند 150 درجة فهرنهايتية ، يتوقف المرحل ويتوقف تيار , يمكنك تغيير درجة الحرارة في السطر 27 :

```
(temp >= 150)
```



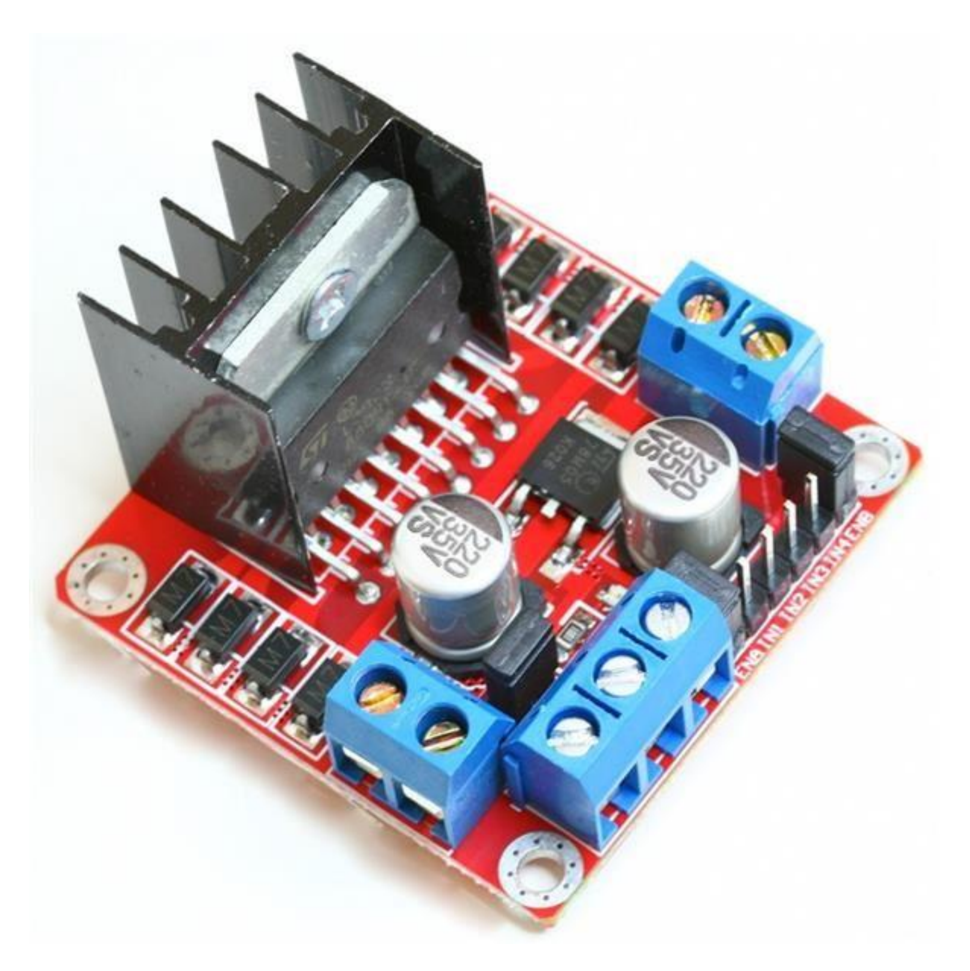

حسنا , في الصورة اعلاه مخطط توصيل لدائرة باستخدام الاردوينو نستطيع الفهم من المخطط انه للتحكم في محركات ال DC باستخدام عصا التحكم Joystick لكن ما هذه القطعة في المنتصف التي يرمز لها L298N ؟

انها نوع من المشغلات drivers الذي يستخدم للتحكم في المحركات بحيث يأخذ الاوامر من الاردوينو , ان الهدف الاساسي من هذه القطعة او المشغلات بشكل عام هي توفير خصائص غير متوفرة في الاردوينو , مثلا اذا اردنا تغذية المحرك ب 12 V لا يمكن لان الاردوينو يعطي 5فولت في حال اذا كان من النوع اونو .

يوجد انواع عدة من المشغلات وهي اساسا عبارة عن دوائر الكترونية من الترانسزتورات والدوائر المتكاملة .. الخ ولكل منها استخدام خاص ووظائف محددة ووحدات تغذيها من الاردوينو او وحدات خارجية ولكن جميعها تشترك بصفة هي انه يتم التحكم بها باستخدام الاردوينو .

بعض انواع المشغلات التي يمكن استخدامها :

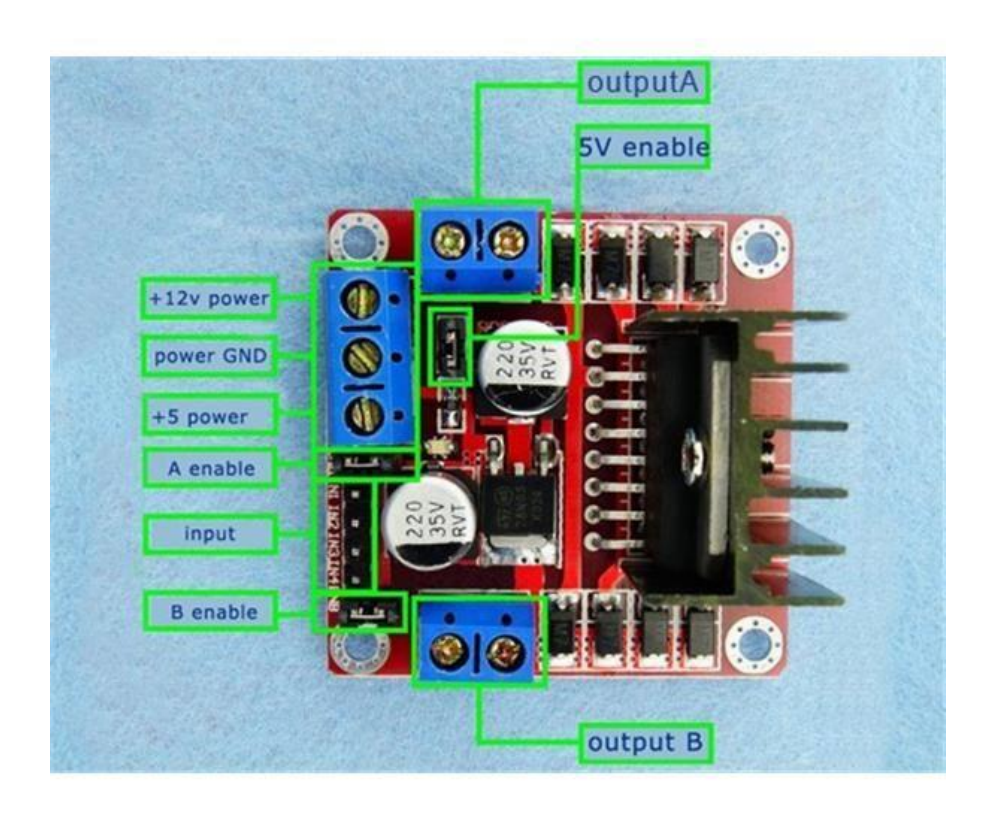
DUAL H-BRIDGE DC & STEPPER MOTOR DRIVER



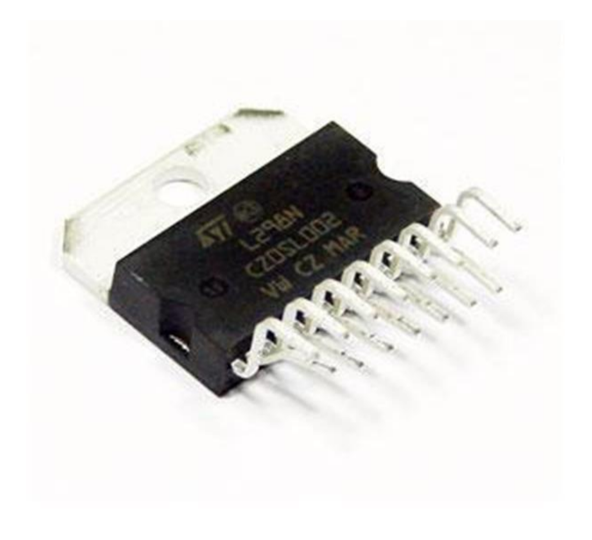

تعمل وحدة التحكم هذه في المحرك الخطوي وعلى تسهيل قيادة إما محركين DC . هذه لوحة ذات جودة عالية جدا وهي صغيرة جدا للتصاميم حيث تكون المساحة مهمة.

ملاحظة: تحتوي هذه الوحدة على مزود طاقة V5 مدمج للاستخدام الخارجي. جهد المحرك هو 7-35 V.

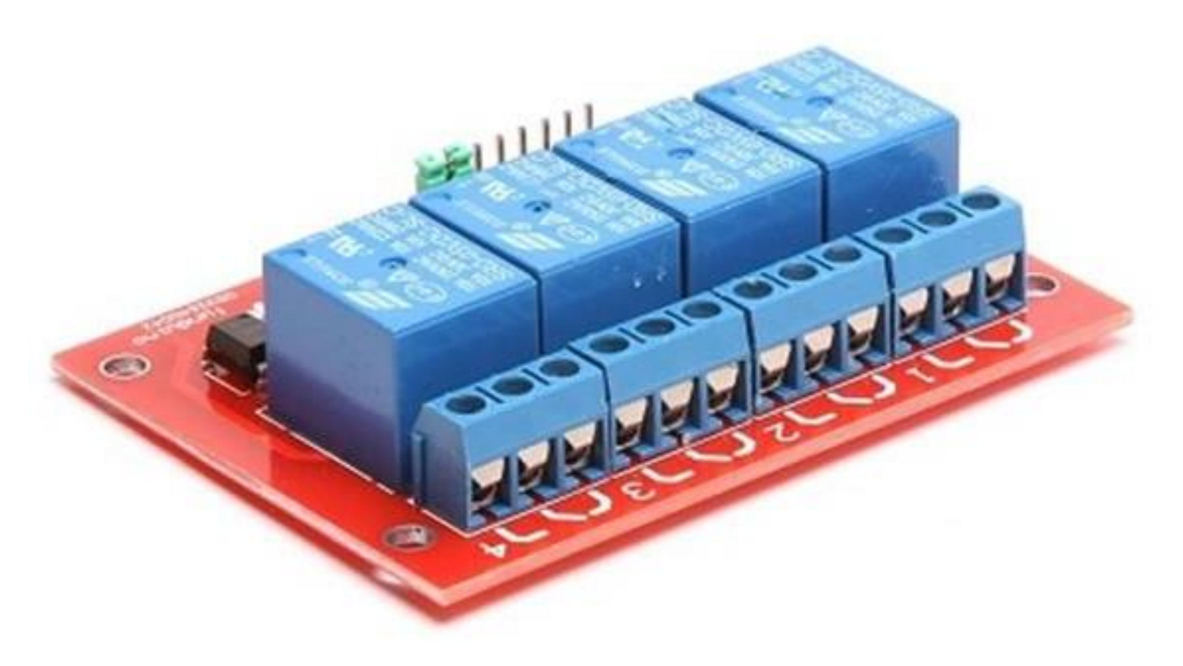
توضيح مكونات الدائرة :




ولكن يمكن ان تاتي هذه الوحدة باستخدام المتحكم الاساسي فقط , فقد تكون بهذا الشكل:



RELAY MODULE 5V DC 4-CHANNEL



هذا المشغل يبدو رائعا , فهو يدعم  التحكم في 4 مرحلات في نفس الوقت !

وخصائصه ما يلي :

2 مصباح  LED لحالةPCB

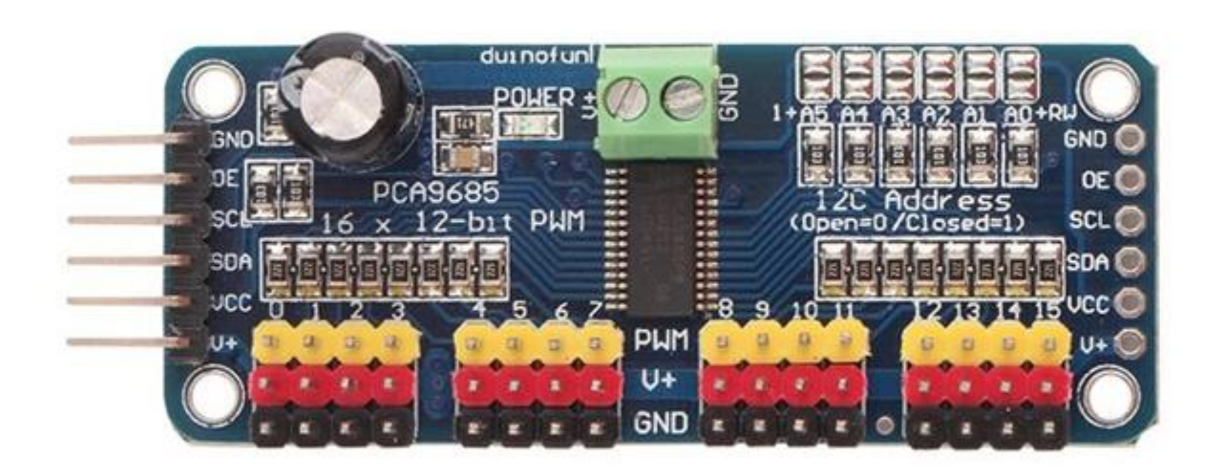
التحكم في AC 250V 10A و DC 30V10A

امدادات الطاقة: يدهم  5 V

منتج يعمل مع الاردوينو اونو ويعمل مع لوحات اردوينو الرسمية.

يوجد منه لونين أحمر و أزرق

# 16-CH SERVO MOTOR CONTROLLER I2C




هذا المشغل يستطيع التحكم في 16 محرك سيرفو في نفس الوقت - وهذه اللوحة بها تشغيل PWM يتم التحكم فيها بواسطة i2c مع ساعة مدمجة.

وخصائصه ما يلي :

منتج يعمل مع الاردوينو اونو ويعمل مع لوحات اردوينو الرسمية.

حوالي 1.6 KHzتردد تعديل PWM

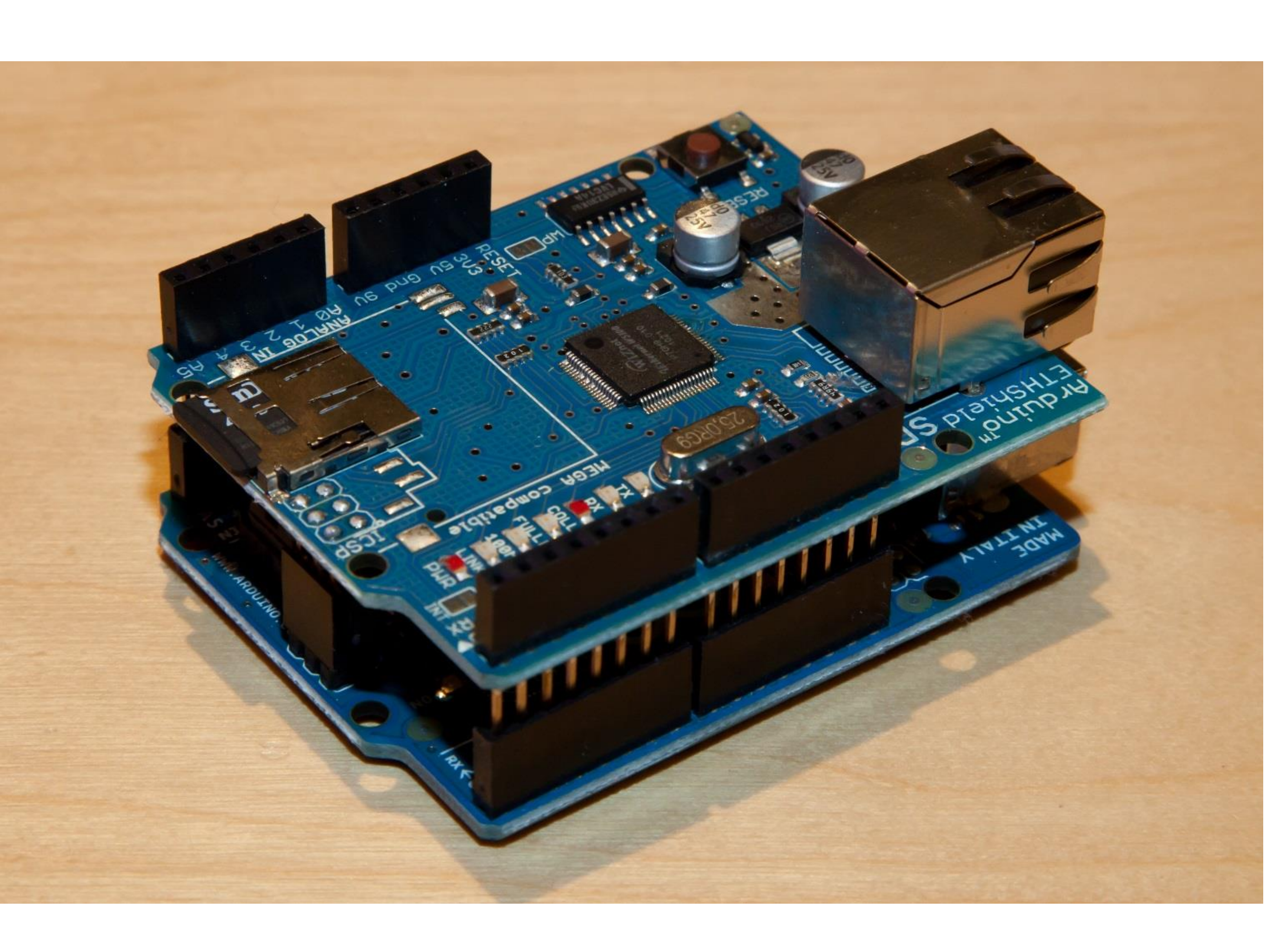
الحصول على خرج دقة 12 بت جاهز للموتور ، مما يعني أن معدل التحديث 60 هرتز يمكن أن يصل إلى دقة us.4

مع حماية الأقطاب العكسية للإدخال.

ضع مكونات المقاومة 220 أوم في جميع خطوط الإخراج PWM ، وذلك لحمايتها ، ويمكنك بسهولة قيادةLED

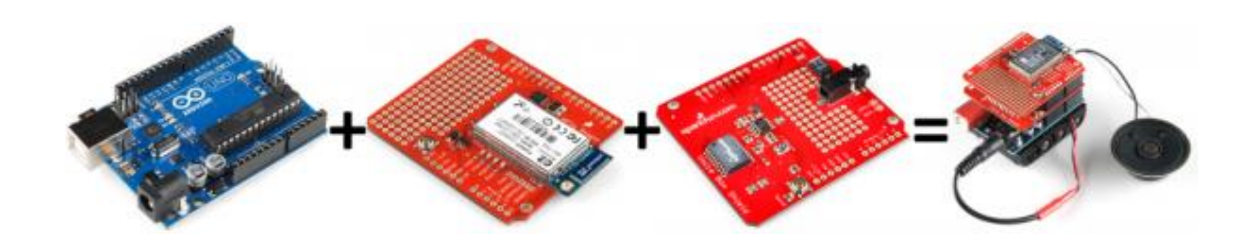
حسنا لتطبيق مشاريع هذا الدرس والدرس القادم , انظر الى الجزء السادس من هذه السلسلة بعنوان (بناء التطبيقات الهندسية )

# الاغطية




أغطية الأردوينو هي ألواح دوائر تركيبية يتم وضعها فوق بطاقات الأردوينو لتزوده بوظائف ومميزات إضافية. هل تريد توصيل بطاقة أردوينو بالإنترنت والتحكم من اي مكان في العالم ؟ هناك غطاء مُخصص لذلك. هل تريد تحويل بطاقة أردوينو إلى سيارة صغيرة؟ هناك أغطية لفعل ذلك. هناك المئات من أغطية الأردوينو موجودة في كل مكان، وهي التي تمنح بطاقات أردوينو إمكانيات مذهلة، بدلاً من أن تكون بطاقات تطويرية جامدة محدودة الإمكانيات.

كثير من أغطية الأردوينو قابلة للتركيب على بعضها البعض. لذلك يُمكننا توصيل الكثير منها معاً لعمل بناء منها. على سبيل المثال يُمكنك توصيل بطاقة أردوينو أونو (Arduino Uno) مع غطاء Voice Box وغطاء WiFly للحصول على جهاز لإصدار الأصوات يتم التحكم فيه لا سلكياً.

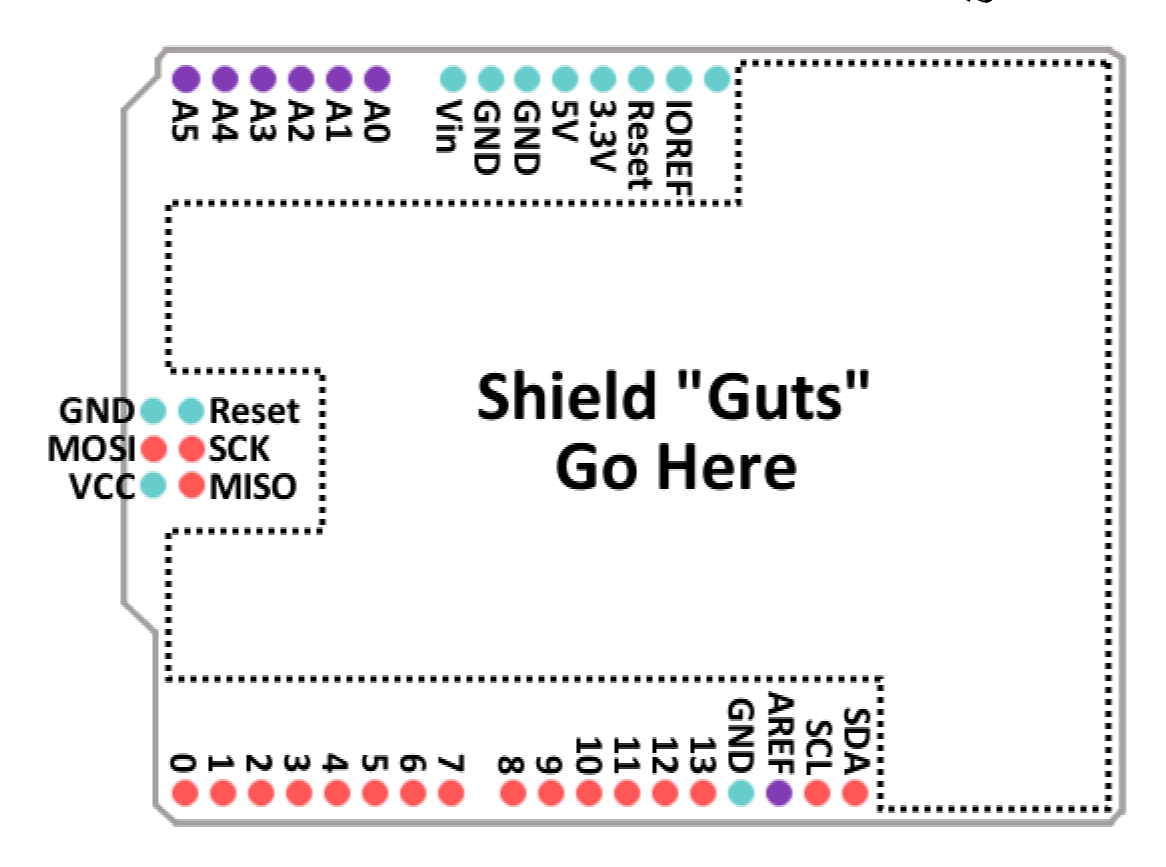


في الغالب تأتي الأغطية مع أمثلة عبارة عن رسومات تخطيطية أو مكتبات. لذلك فبساطة استخدامها لا تقتصر على تركيبها ببطاقات أردوينو فقط، بل كل ما تحتاجه لتجعل تلك الأغطية تعمل هو أن تقوم يتحميل أحد الأكواد / المكتبات المرفقة إلى بطاقة الأردوينو.

## معامل شكل الغطاء (Shield Form Factor)

أي غطاء من أغطية أردوينو يجب أن يكون له معامل شكل مطابق لمعامل الشكل الخاص ببطاقات الأردوينو القياسية. منافذ الطاقة والأرضي (ground) عبارة عن وصلة رأس مسنن (pin header) ذات ثمانية منافذ (في السابق كانت ذات ستة منافذ)، والمنافذ التناظرية عبارة عن وصلة رأس مسنن ذات ستة منافذ مجاورة للسابقة. أما المنافذ الرقمية فتوجد على الجانب الآخر من اللوح وهي عبارة عن وصلة رأس مسنن ذات ثمانية منافذ يفصل بينها وبين وصلة رأس مسنن أخرى ذات عشرة منافذ مسافة تساوي نصف بوصة. بعض الأغطية تتطلب توصيلها بمنافذ ICSP على لوح الأردوينو (منافذ خاصة بالبرمجة في نهاية البطاقة عبارة عن صفين كل منهما يحتوي على ثلاثة منافذ).

بعض الأغطية تستخدم جميع المنافذ المتاحة على بطاقات أردوينو، بينما البعض الآخر يستخدم بعض تلك المنافذ فقط. عند تركيب الأغطية من المهم جداً التأكد أن كل سن يدخل في المنفذ الخاص به وأنه لا يحدث تداخل بين المنافذ. بعض الأغطية تتصل بالأردوينو عبر تقنيات SPI أو I2C أو Serial، بينما البعض الآخر يعتمد على منافذ الدخل التناظرية.



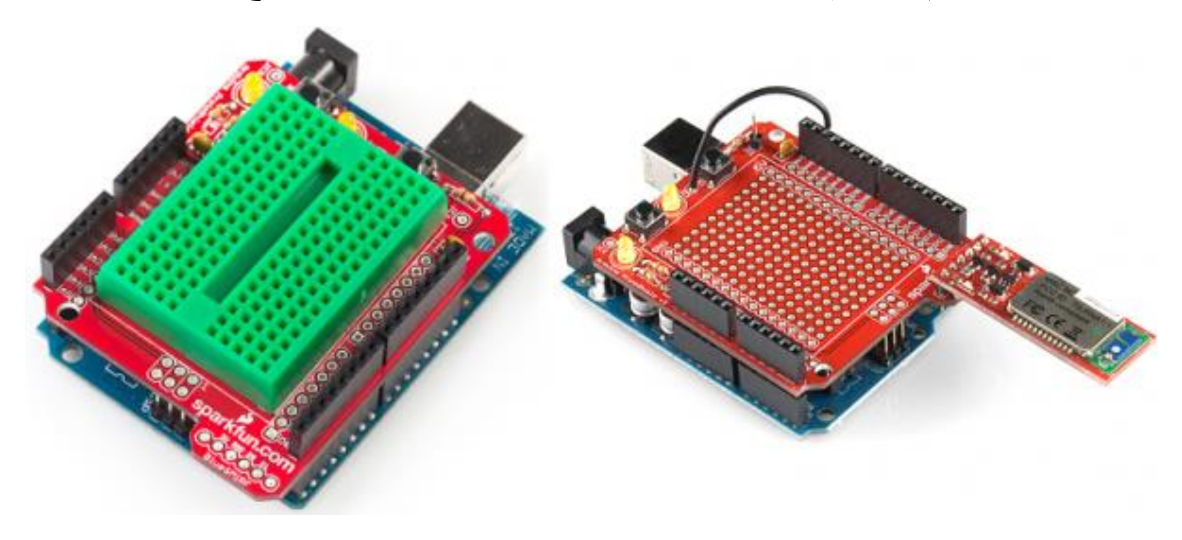

هناك الكثير والكثير من أغطية الأردوينو متاحة للاستخدام، لذلك من الصعب أن نغطيها جميعاً في هذا الدرس. في الجزء التالي سنذكر مجموعة من أكثر الأغطية تفرداً وشعبية.

هذه  قائمة بأكثر الأغطية تميزاً وشيوعاً. هذه القائمة ليست شاملة لجميع الأغطية الموجودة يُمكنك العثور على قائمة كهذه من shieldlist.org

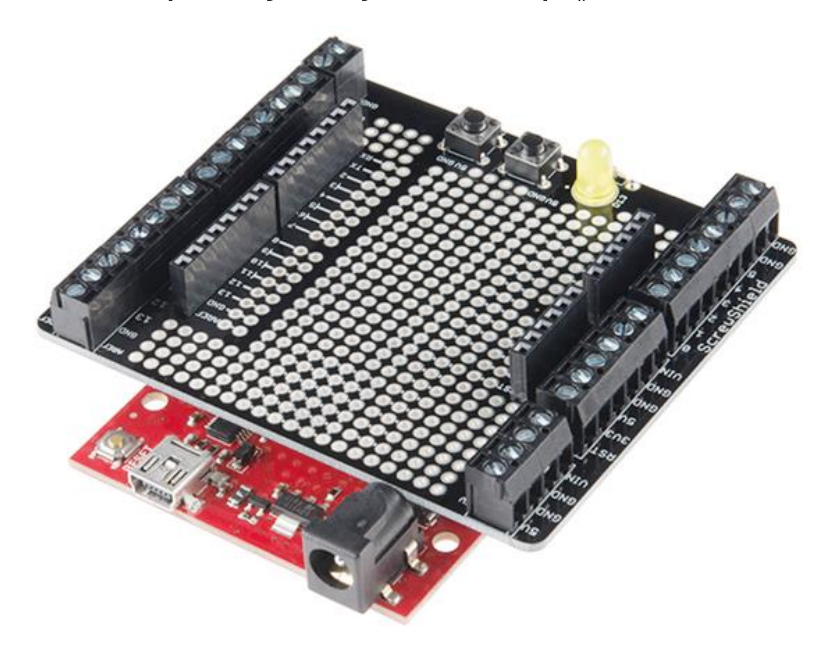
## أغطية عمل النماذج الأولية (Prototyping)

لا تضيف أغطية النماذج الأولية الكثير من الوظائف إلى بطاقات أردوينو، ولكنها تُساعدنا بطريقة أخرى. هذه الأغطية بشكل عام تجعل من عمليات التوصيل من وإلى بطاقات أردوينو أسهل كثيراً، فمن الممكن مثلاً الاستعانة بها لتحويل منافذ البطاقة إلى نوع وصلات البراغي (screw terminals) :

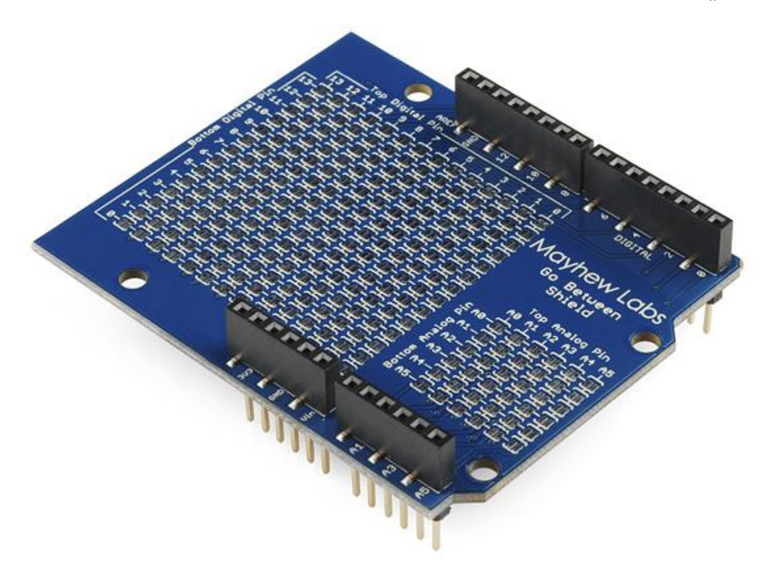
غطاء ProtoShield Kit: نجم هذه الفئة بلا منازع. هذا الغطاء هو عبارة عن مساحة كبيرة لعمل النماذج الأولية. يُمكنك أن تثبت أعلاه لوح تجارب (breadboard) صغير أو أن تقوم باللحام مباشرة على المساحة الخاصة بعمل النماذج الأولية.



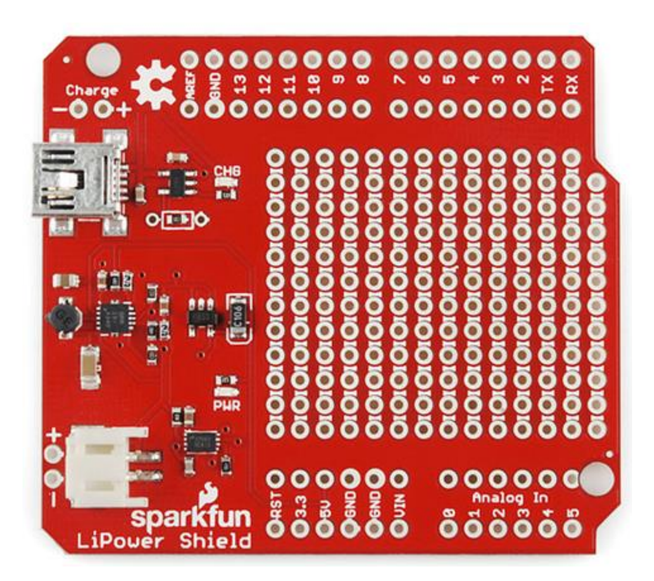
غطاء :ProtoScrew Shield مثل غطاءProtoShield ، لكن يتم تحويل كل منفذ إلى منفذ براغي .(screw terminal) وهو مفيد للغاية عند توصيل مُحركات خارجية أو حساسات أداء عالي.(heavy-duty sensors)



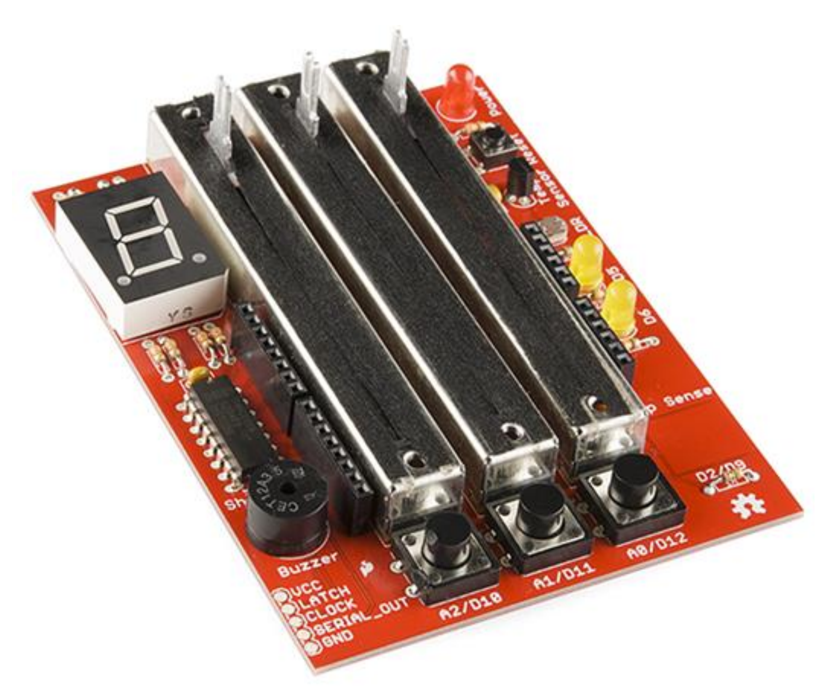

غطاء :Go-Between Shield وظيفة هذا الغطاء هو أن يوضع بين غطاءين آخرين. حيث يقوم بإلغاء بعض منافذ الغطاء الموجود في الأعلى لكيلا تتداخل مع منافذ الغطاء الموجود في الأسفل.



غطاء :LiPower Shield هذا الغطاء يسمح لك بتوصيل الطاقة إلى لوح أردوينو الخاص بك باستخدام بطارية الليثيوم بوليمر.



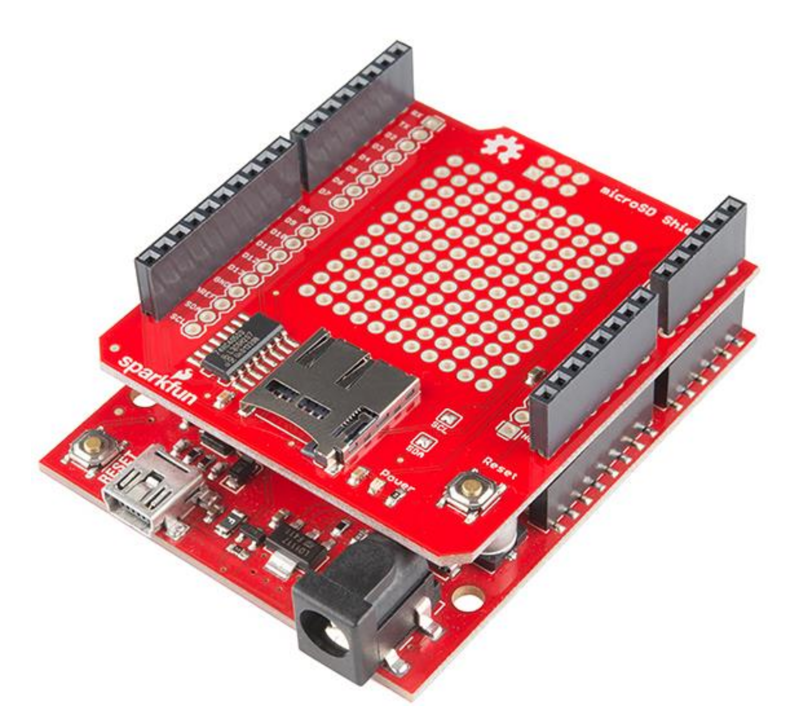

غطاء :Danger Shield أروع الأغطية على الإطلاق. هذا الغطاء عبارة عن خليط عجيب من الشاشات ومقاييس الجهد الانزلاقية (potentiometer) وحساسات أخرى. وهو غطاء رائع للتعلم على بطاقات أردوينو ولمشاريع المؤثرات الصوتية.



غطاء Joystick Shield Kit: هذا الغطاء يجعل من بطاقة أردوينو جهاز تحكم بدائي. حيث يحتوي على عصا توجيه (joystick) وأربعة أزرار، مما يجعله رائعاً للتحكم في الروبوتات.

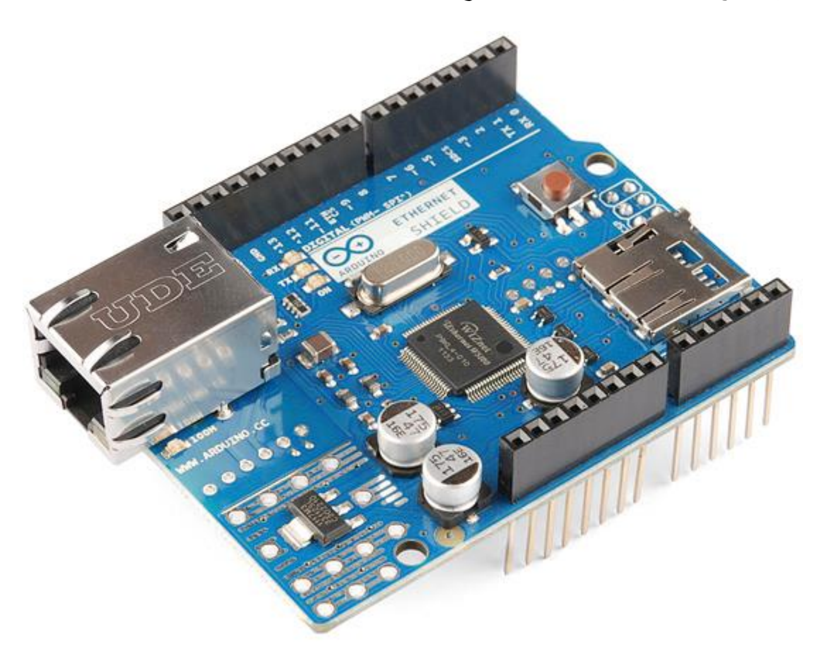



غطاء microSD Shield: تحتوي بطاقات أردوينو على سعة تخزين محدودة، لكن هذا الغطاء سهل الاستخدام (مع استخدام مكتبة SD) يتيح إضافة المزيد من سعة التخزين.



## أغطية الإيثرنت (Ethernet) والوايفاي (WiFi) والاتصال اللاسلكي (Wireless) ونظام تحديد المواقع (GPS)

غطاء الإيثرنت :Arduino Ethernet Shield يُعد هذا الغطاء أحد الأغطية الكلاسيكية. حيث يمنح غطاء الإيثرنت هذا بطاقة الأردوينو القدرة على الاتصال بشبكة الإنترنت العالمية. وهناك أيضاً مكتبة رائعة تدعمه.



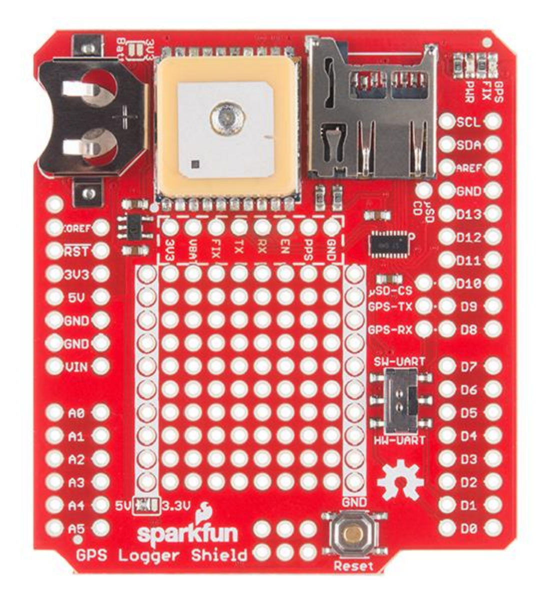

غطاء :WiFly Shield الغطاء الأساسي الخاص بتقنية WiFi لدى .SparkFun يمنح هذا الغطاء بطاقات أردوينو القدرة على الاتصال بالشبكات اللاسلكية العاملة ببروتوكولات 11 .b/gوبالتالي يمكن استخدامها كخادم (server) أو كعميل (client)أو كلاهما .



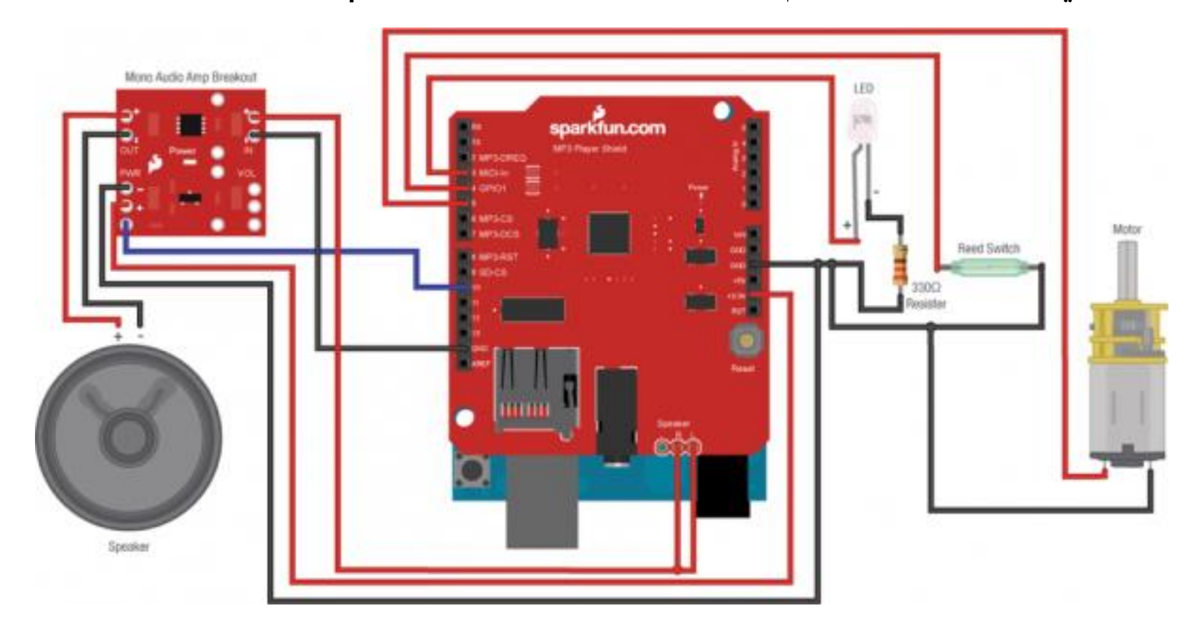
غطاء Cellular Shield w/ SM5100B: بإمكان هذا الغطاء تحويل بطاقة الأردوينو الخاصة بك إلى هاتف خلوي! قم بإرسال رسائل SMS أو قم بتركيب سماعة وميكروفون لاستخدامه لإجراء المكالمات واستغني عن هاتفك المحمول.



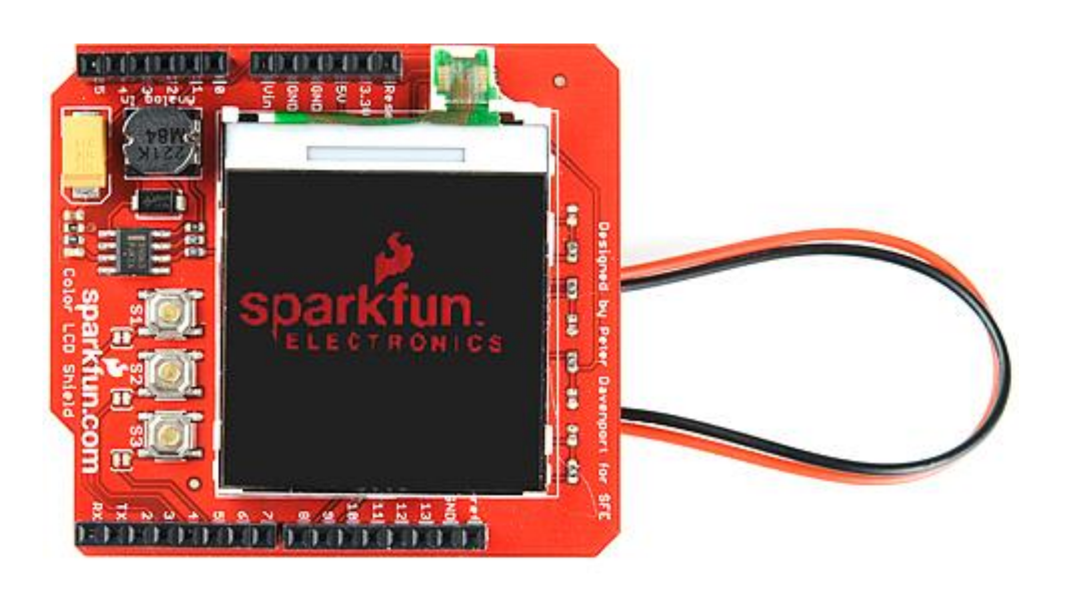

غطاء GPS Shield: نظام الملاحة العالمي (GPS) ليس مُعقداً بالدرجة التي يمكن ان تتخيلها. فباستخدام غطاء GPS يُمكن للوح الأردوينو تحديد موقعه على الدوام.



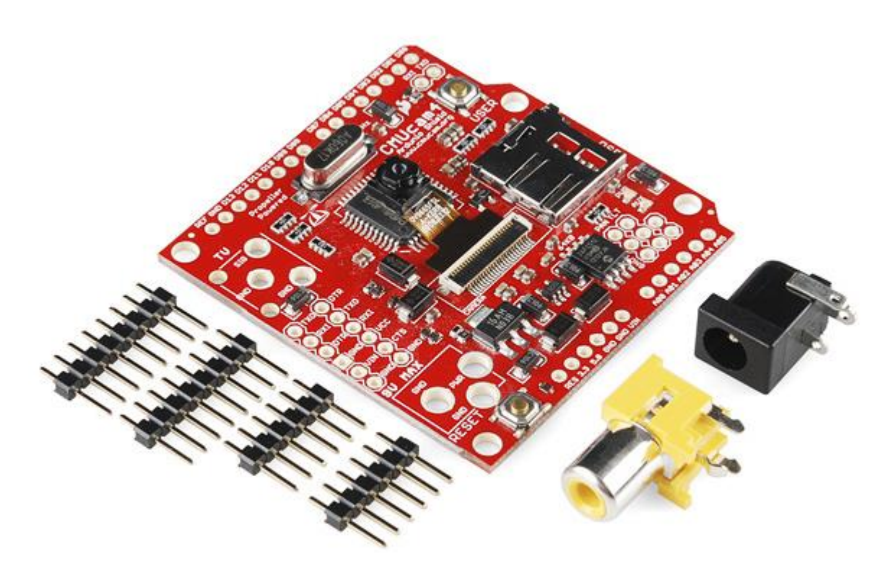
غطاء MP3 Player Shield: قم بتحويل بطاقة أردوينو الخاصة بك إلى مُشغل mp3. كل ما عليك فعله هو إدخال بطاقة ميكرو SD وإضافة سماعات وتحميل الكود البرمجي المُخصص، ومن ثم تحصل على مُشغل موسيقى mp3 خاص بك.



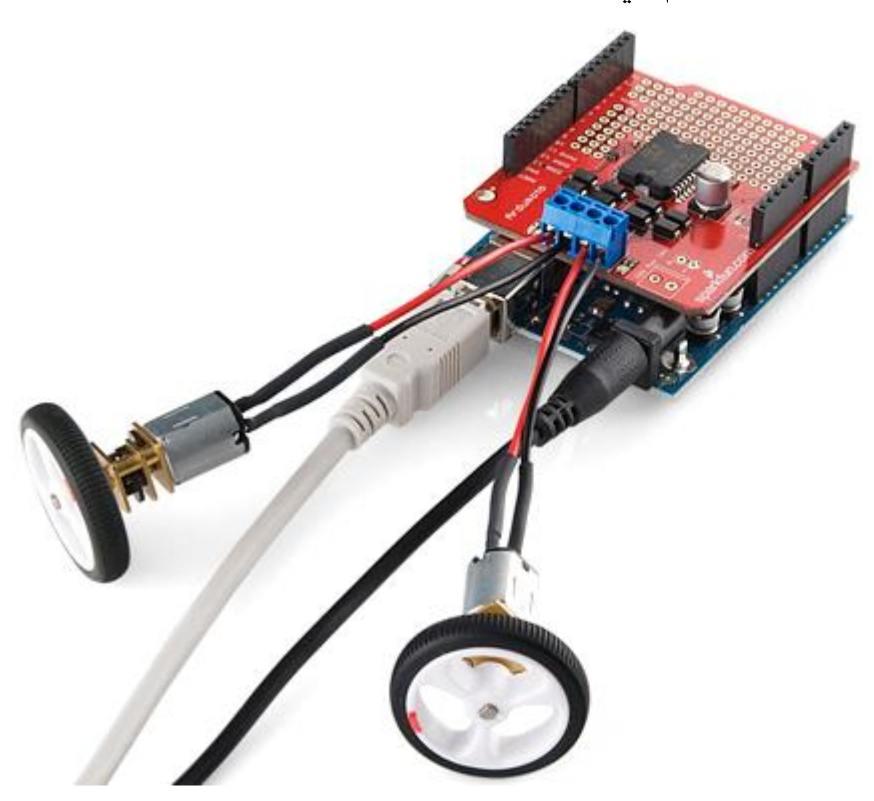

غطاء Color LCD Shield: زود بطاقة أردوينو خاصتك بشاشة LCD ملونة فريدة مثل المستخدمة في الهواتف النقالة بدقة 128×128.



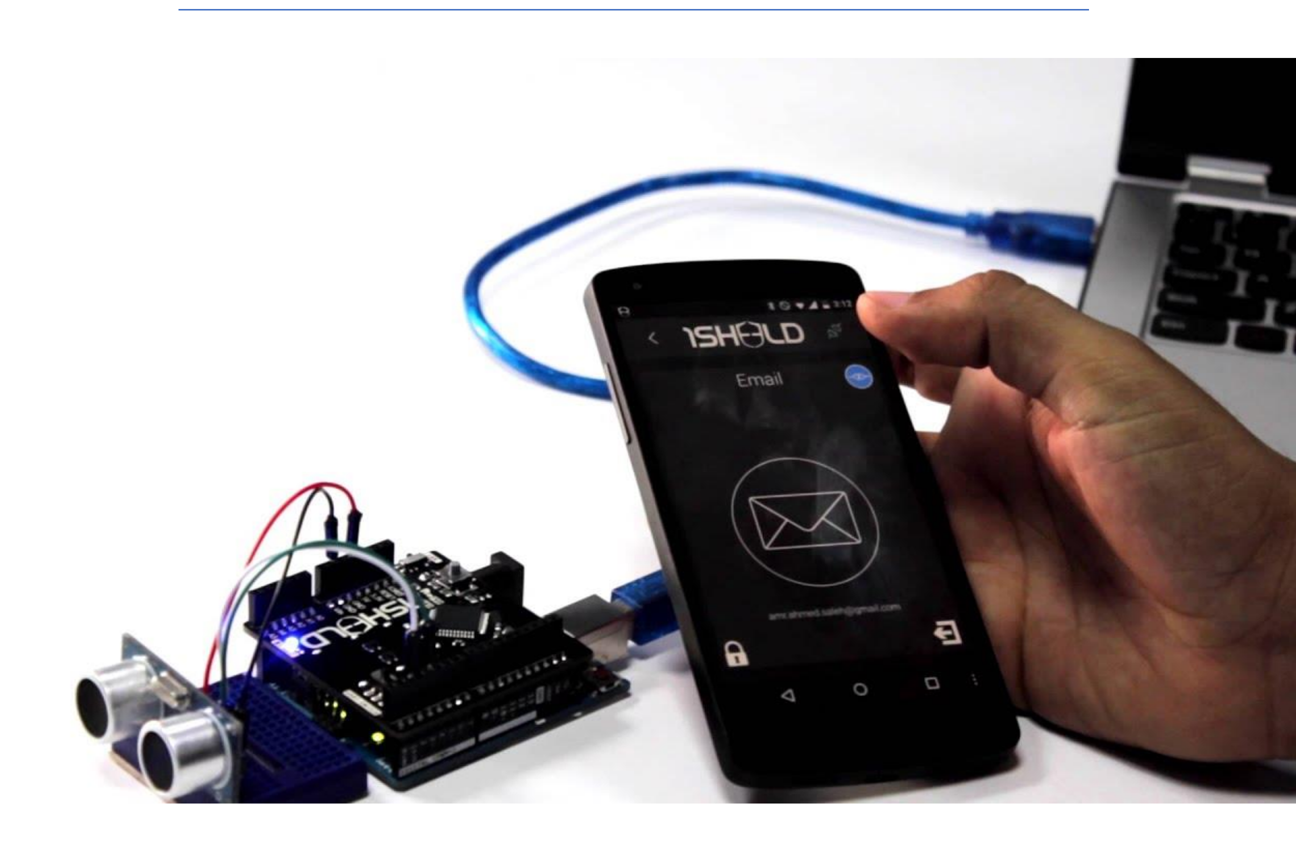
غطاء :CMUcam هذه الكاميرا تمنح بطاقة أردوينو خاصتك القدرة على الرؤية. يُمكنك الاستعانة بها لمراقبة طريق الروبوت لتفادي أية عقبات.



غطاء :Ardumoto Motor Driver Shield غطاء كلاسيكي لتشغيل المحركات، ويُمكن من خلاله التحكم في محركين.DC



# برمجة الاردوينو باستخدام الهاتف




تعلمنا في الدروس السابقة كيفية استخدام هاتف يعمل بنظام اندرويد للتحكم في الاردوينو , لكن في هذا الدرس سوف نستخدم جهاز الهاتف كمنصة لكتابة الكودات البرمجية وتحميل الكود للاردوينو , سابقا كنا دائما نعمل على بيئة IDE للبرمجة لكن في هذه السلسلة سوف نتعلم برمجة الاردوينو باكثر من طريقة منها الهاتف وبرنامج الماتلاب وال Lab View وذلك في الجزء السادس كتاب (بناء التطبيقات الهندسية).

يمكننا برمجة اردوينو باستخدام هاتفك الذكي. لانه في بعض الأحيان ليس لدينا أي جهاز كمبيوتر أو كمبيوتر محمول لبرمجة لوحات Arduino الخاصة بنا. لا يزال بوسعنا أن نبرمجها باستخدام جوال Android ، بفضل الكيبل OTG On the Go

قد تكون استخدمت كيبل OTG لتوصيل أجهزة التحكم في الألعاب ، وإعطاء الطاقة للأجهزة الصغيرة او ما شابه . يمكنك القيام بالكثير من الأشياء الأخرى ويمكن تشغيل لوحة Arduino مع الهاتف الذكي باستخدامها ، سنقوم بتجميع وتحميل كود اردوينو باستخدام تطبيق أندرويد يسمى "ArduinoDroid" وهو مماثل تماما ل Arduino IDE

القطع المطلوبة :

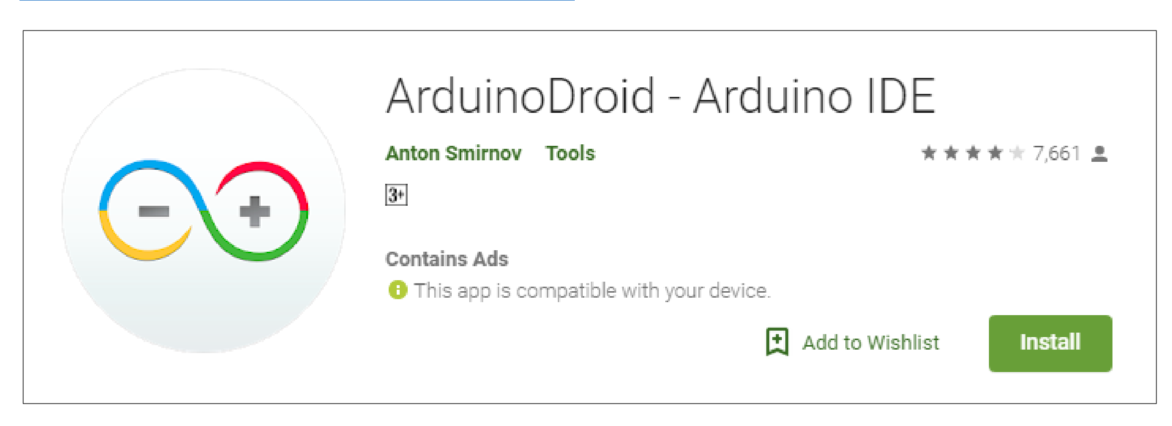
1. Arduino Board
2. OTG cable



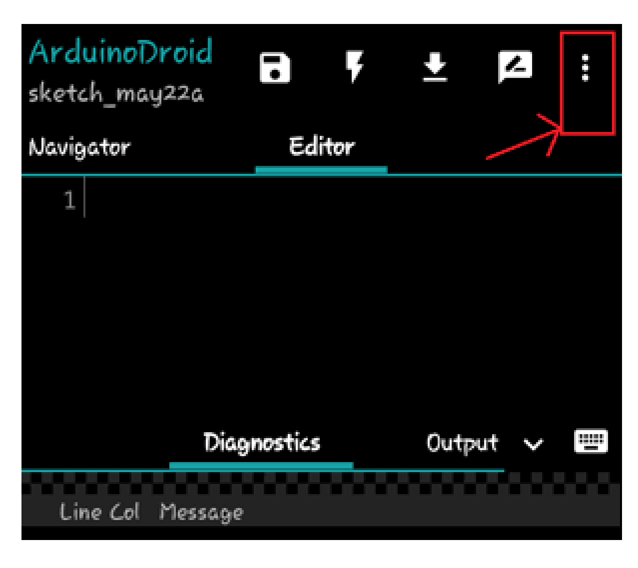
3. Arduino USB cable
4. جهازAndroid

الخطوة 1: قم بتنزيل التطبيق من الرابط الموضح أدناه أو ببساطة اذهب إلى متجر Play وابحث عن ArduinoDroid وقم بتثبيته.

https://play.google.com/store/apps/details?id=name.antonsmirnov.android.arduinodroid2



الخطوة 2: افتح التطبيق بعد التثبيت. سيبدو كما هو موضح أدناه:



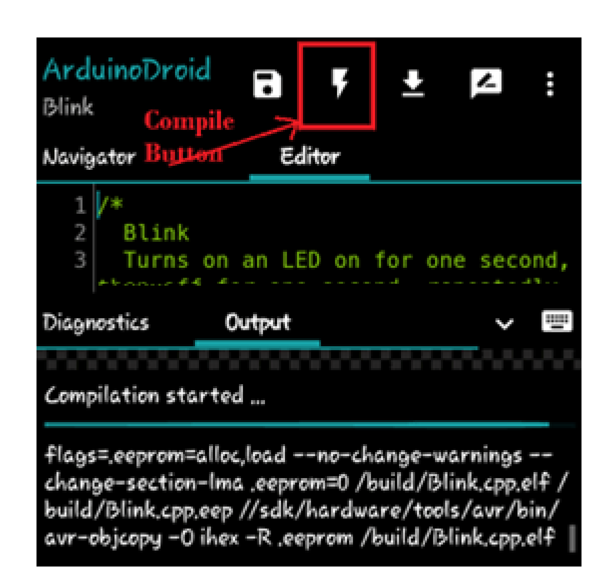

في هذه النافذة ، يمكنك كتابة الكود الخاص بك أو ببساطة الحصول على امثلة من القائمة (تظهر من خلال ثلاث نقاط في الزاوية اليمنى العليا)

الخطوة 3: قم بتوصيل لوحة Arduino باستخدام جهاز Android باستخدام كابل USBو OTG.

الخطوة 4: اختر اللوحة الخاصة بك من.Setting> Board Type

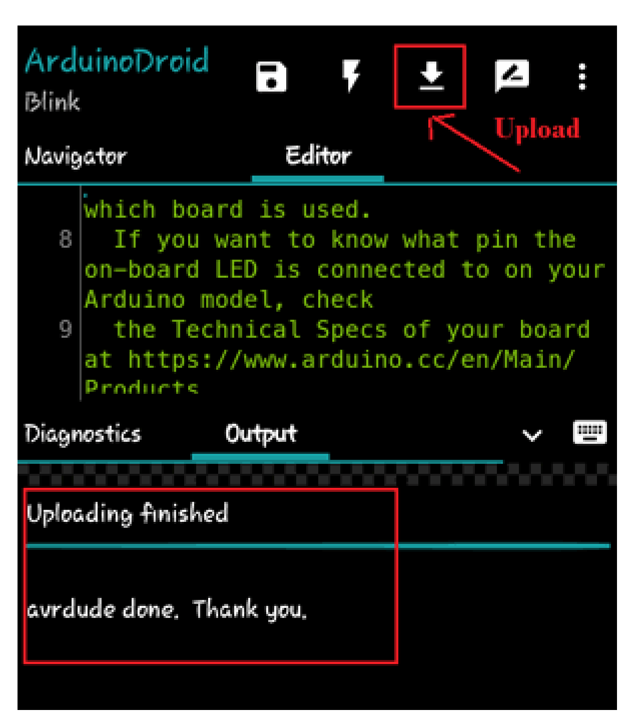
في Arduino IDE ، إذا ضغطنا على زر upload ، فإن برنامجنا يتم الترجمة compile أولاً ومن ثم يتم تحميله upload

ولكن هنا علينا أن نجمع compile أولاً بالنقر فوق زر الترجمة Compile كما هو موضح أدناه.

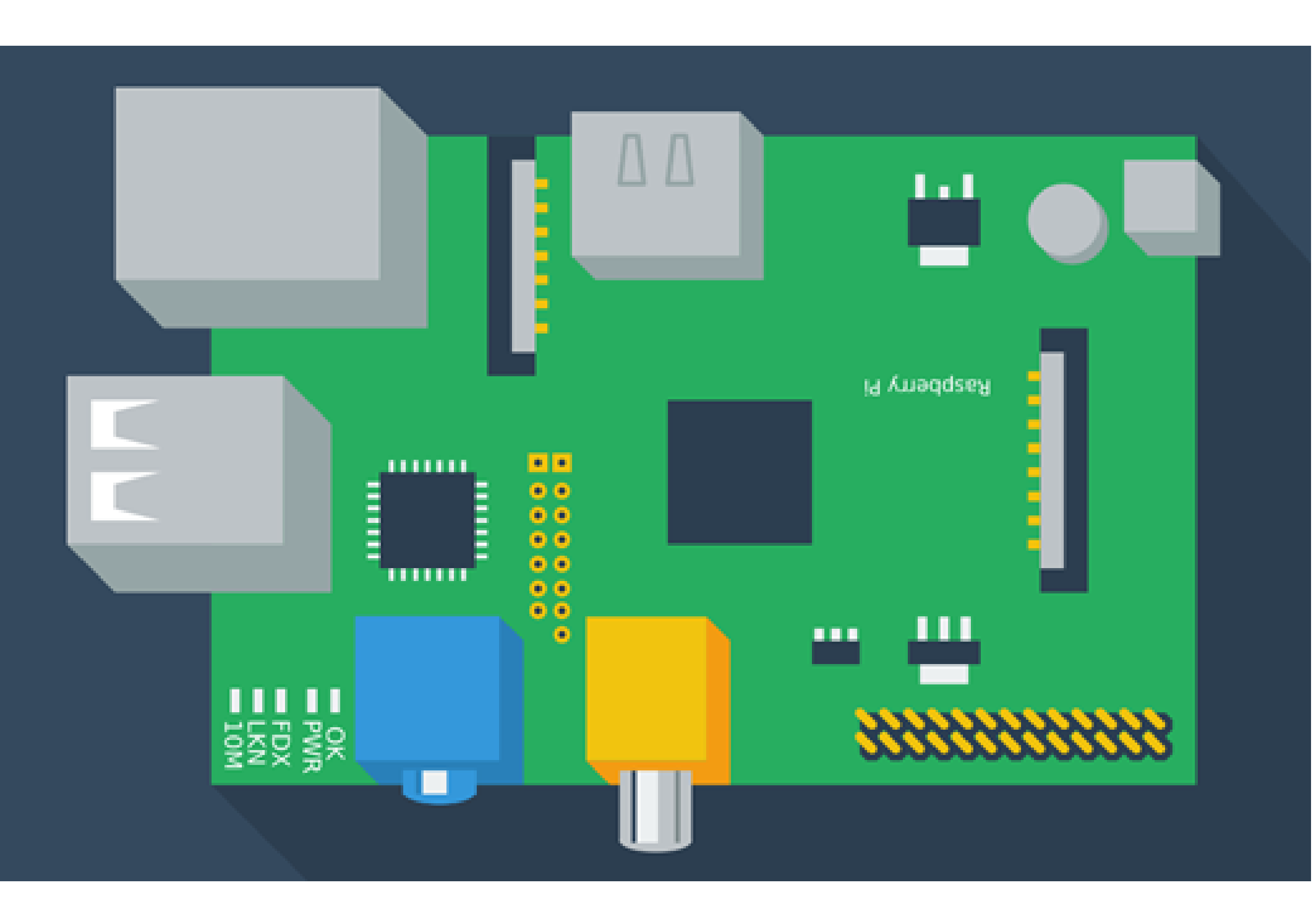



يمكنك رؤية حالة التجميع في نافذة الإخراج.

الخطوة 5: بمجرد اكتمال التحويل البرمجي ، انقر فوق الزر Upload "تحميل" كما هو موضح أدناه , يمكنك الآن البرمجة بواسطة هاتفك دون الحاجة الى حاسوب .

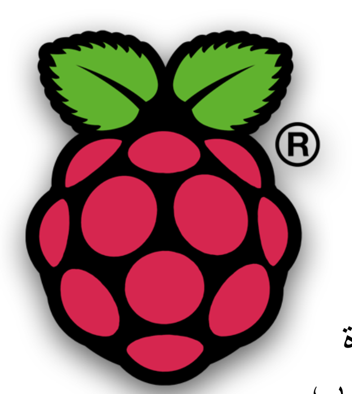


## مقدمة في الراسبيري باي




راسبيري باي هو كمبيوتر متكامل صغير في حجم كف اليد او بحجم بطاقة الائتمان استهلاكه للطاقة أقل من 5 وات.

تم تصميمه في جامعة كامبريدج للمساعدة في تدريس علوم الحاسب للطلاب بها بحيث يدمج بتقنيته ما بين أنظمة لينكس وعلم البرمجة والإلكترونيات وأنظمة التحكم الذكية في ذات الوقت!، مما جعل هذا الكمبيوتر الصغير يحقق نجاح مذهل في كلا المجالين التعليمي والتطبيقي.



مصنوع من شريحة الكترونية المعروفة باسم نظام SOCأو System On Chip يحتوي هذا الحاسب على معظم المنافذ التي يمتلكها اي حاسب آلي بل قد تزيد يحتوي الحاسب على معالج رسوميا GPU ثنائي النواة قادر على تشغيل أفلام عالية الدقة HD وذاكرة عشوائية بحجم يبدأ ب 512 ميجابايت بالإضافة إلى المنافذ الكثيرة التي يملكها . يمتلك الحاسب منفاذ USB لادخال الفأرة او لوحة المفاتيح او اي جهاز من كماليات جهاز الحاسوب ومنفذ Ethernet لشبكات الحاسب والانترنت

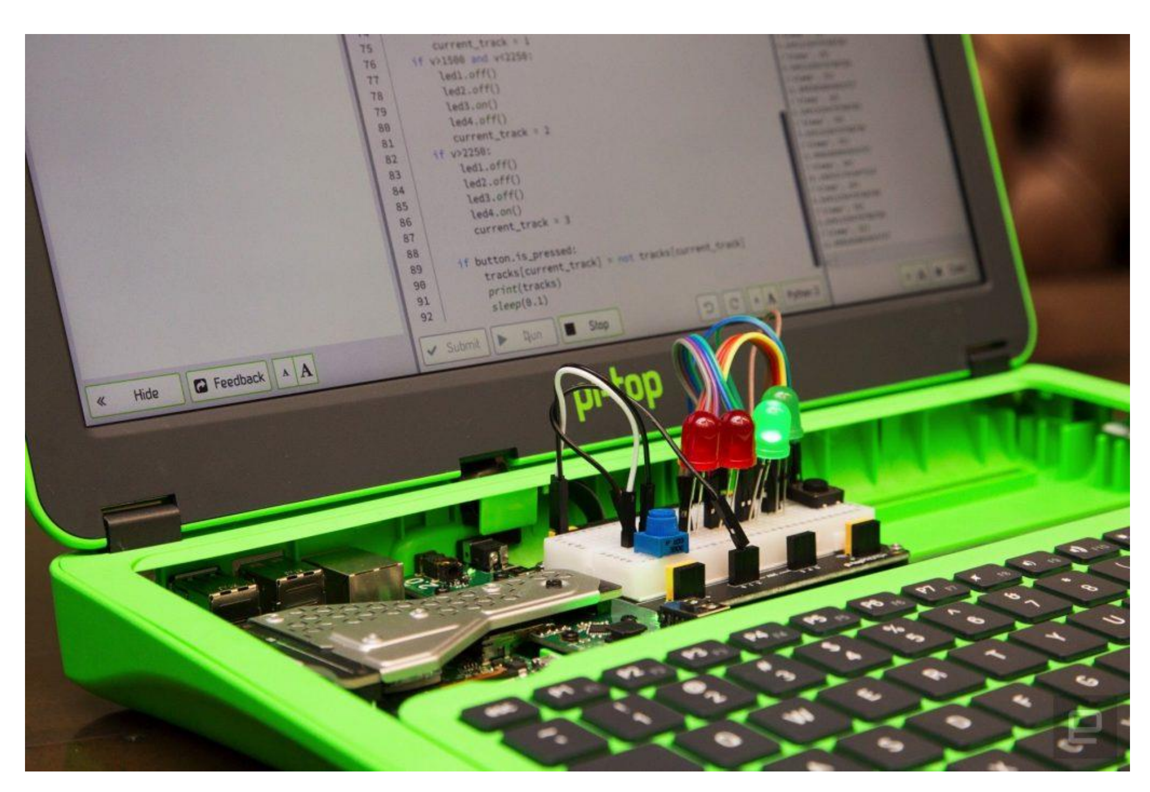
ومنفذ HDMI بالإضافة إلى منفذ video للتلفزيون ومنفذ لذاكرة SD Card يمتلك أيضا منفذ للشاشات الصغيرة يتيح لك استخدام المهام المختلفة كبرامج التحرير والتصفح وحتى الألعاب والتحكم في الاجهزة ومنافذ كثيرة للتحكم الرقمي كالتي موجودة في المتحكمات الدقيقة Microcontroller وايضا نفذ Micro USB للطاقة وكاميره رقمية عالية الجودة .

- ماذا يمكنني الفعل براسبيري باي ؟

يمكنك استخدام الراسبيري كأي حاسب آلي تقليدي لتصفح الانترنت وارسال البريد الالكتروني وحتى تحرير الملفات والوثائق أيضاً تستطيع تحويل أي تلفاز عندك إلى نظام ترفيه منزلي متصل بالإنترنت وكذلك يمكنك عمل مشاريع تحكم إلكترونية مذهله واستخدام الراسبيري كبديل متطور جداً عن المتحكمات الدقيقة "Micro controllers" فمثلا يمكنك عمل التطبيقات التالية:

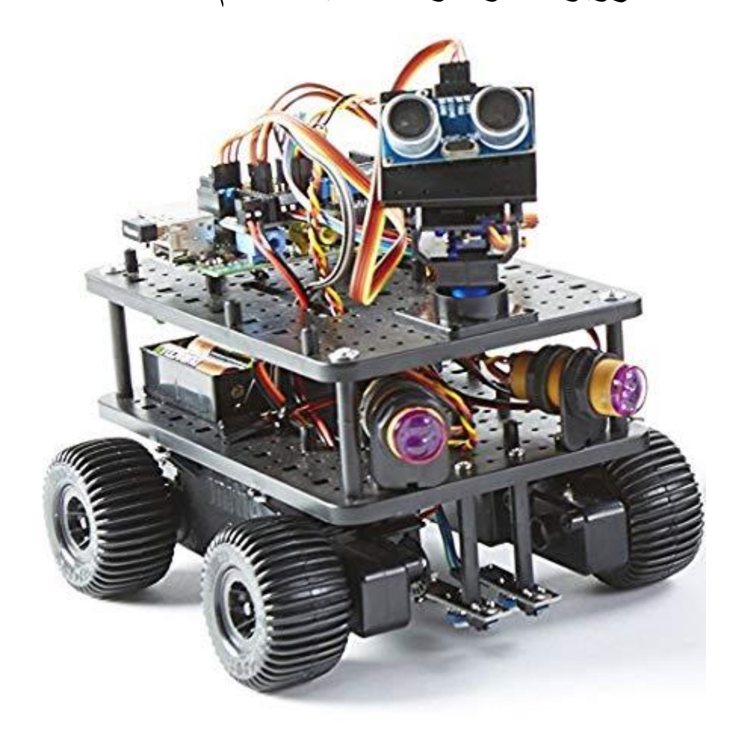
- تصميم نظم التحكم الخاصة بالمنازل الذكية
- تطبيقات المراقبة مثل عمل كاميرات لبث الفيديو والصور عن بعد والمراقبة البيئية مثل عمل نظام لمراقبة درجات الحرارة والرطوبة عن بعدRemote Monitor
- التلفاز الذكيSmart TV
- خوادم لينكس المختلفة مثل HTTP, FTP, SSH, VPN, MySql
- الحواسيب الفائقة Supercomputers

# حاسب محمول

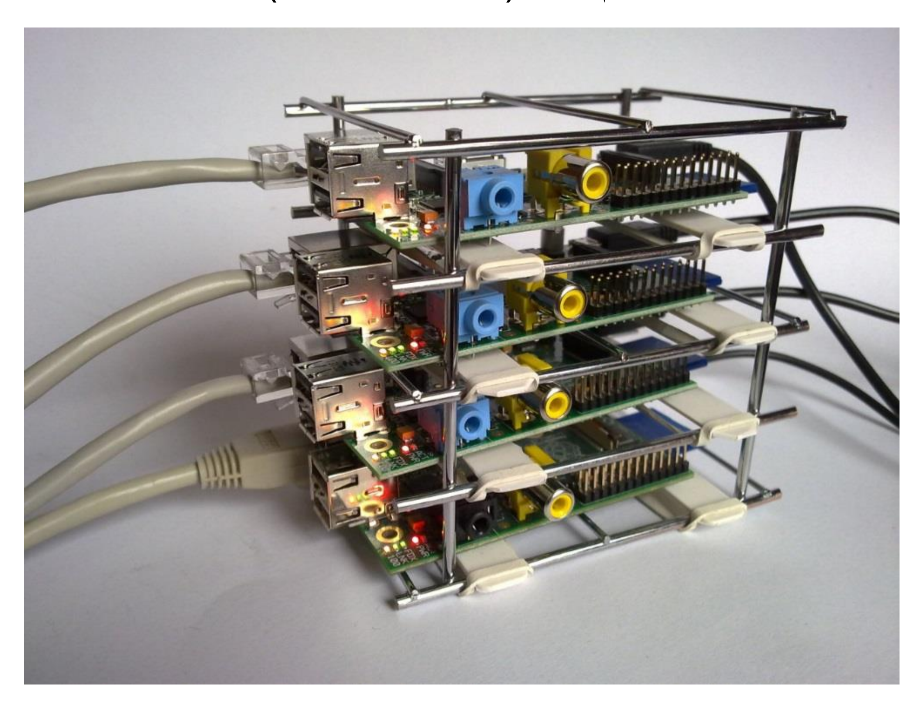



قلنا في البداية أنّ راسبيري باي عبارة عن حاسب بكل ما تحمله الكلمة من معنى، ولكن للدقة يجب أن نقول أنّها تمثل اللوحة الأم الّتي تحوي المعالج والذواكر وبطاقة الرسوميات. لذلك، ومع إضافة بعض المكونات الأخرى يمكننا الحصول على حاسب محمول متكامل.

إن كنت تظن أنّ الأمر مزحة فصدقني هو ليس كذلك، وحتى تتأكد بنفسك، فإنّني أدعوك للتعرف علىpi-top ، وهو عبارة عن حاسب محمول مبني على راسبيري باي 3. يعمل pi-top بنظام لينوكس، ويحتوي على الكثير من البرامج التعليمية التي تساعد الأطفال على تعلّم البرمجة.



خادم ويب (Web Server)



مسرح منزلي او مشغل العاب




طائرات التحكم عن بعد وطائرات بدون طيارRobots ، ROV and UAV

## *انواع الراسبيري باي*

DIFFERENCE BETWEEN VARIOUS RASPBERRY PI MODELS






---

يوجد العديد من اصدارات الراسبيري باي المتوفرة في الاسواق وتختلف بشكل اساسي في عدد الاطراف سواء كانت منافذ طرفية/مسامير او USB وتختلف ايضا الانواع في حجم الذاكرة العشوائية التي تبدأ ب512 ميجا بايت , الى 2 جيجا وربما أكثر ونوع المعالج الذي ياتي احيانا رباعي النواة بتردد 1.4 غيغا هيرتز ويوجد بعض الانواع بها مخرج HDMI وبعضها لا يحتوي واضافة الى ذلك الخاصية المهمة لدى البعض وهي الاتصال الالسلكي وايرلس , هذا كله فقط في لوحة الكترونية بحجم كف اليد ولا يتجاوز سعرها اجمالا ال 40 دولار!
لكن معظم انواع الراسبيري باي لا يحتوي على زر للتشغيل والاطفاء مثل الحواسيب المعتادة . للتشغيل: اوصل الجهاز فقط! للإيقاف: افصل الكهرباء وسنتعلم تماما كيف نفعل هذه الامور . قطعة الرام مركبة فوق قطعة المنظومة على رقاقة SoC لذا فهي غير قابلة للفك أو التبديل
افتراضياً, الشركة تدعم لغة Python كـلغةٍ تعليمية. لكن أي لغة برمجة أخرى تُجمع على ARMv6 ستكون قابلة للاستخدام.

حسنا للتعرف على اصدارات الراسبيري باي المختلفة انظر الملحق B في اخر الكتاب.
سوف ندرس اللوحة من النوع (Raspberry Pi 2 Model B)




يوفر جهاز Raspberry Pi 2 6 أضعاف قدرة معالجة النماذج السابقة. هذا الجيل الثاني من Raspberry Pi يحتوي على معالج Broadcom BCM2836 الذي تمت ترقيته ، وهو معالج رباعي النواة يعتمد على ARM Cortex-A7 والذي يعمل بسرعة 900 ميجا هرتز. كما يتميز بزيادة في سعة الذاكرة إلى 1 جيجا بايت.

## المواصفات من ورقة البيانات Datasheet

**Specifications**

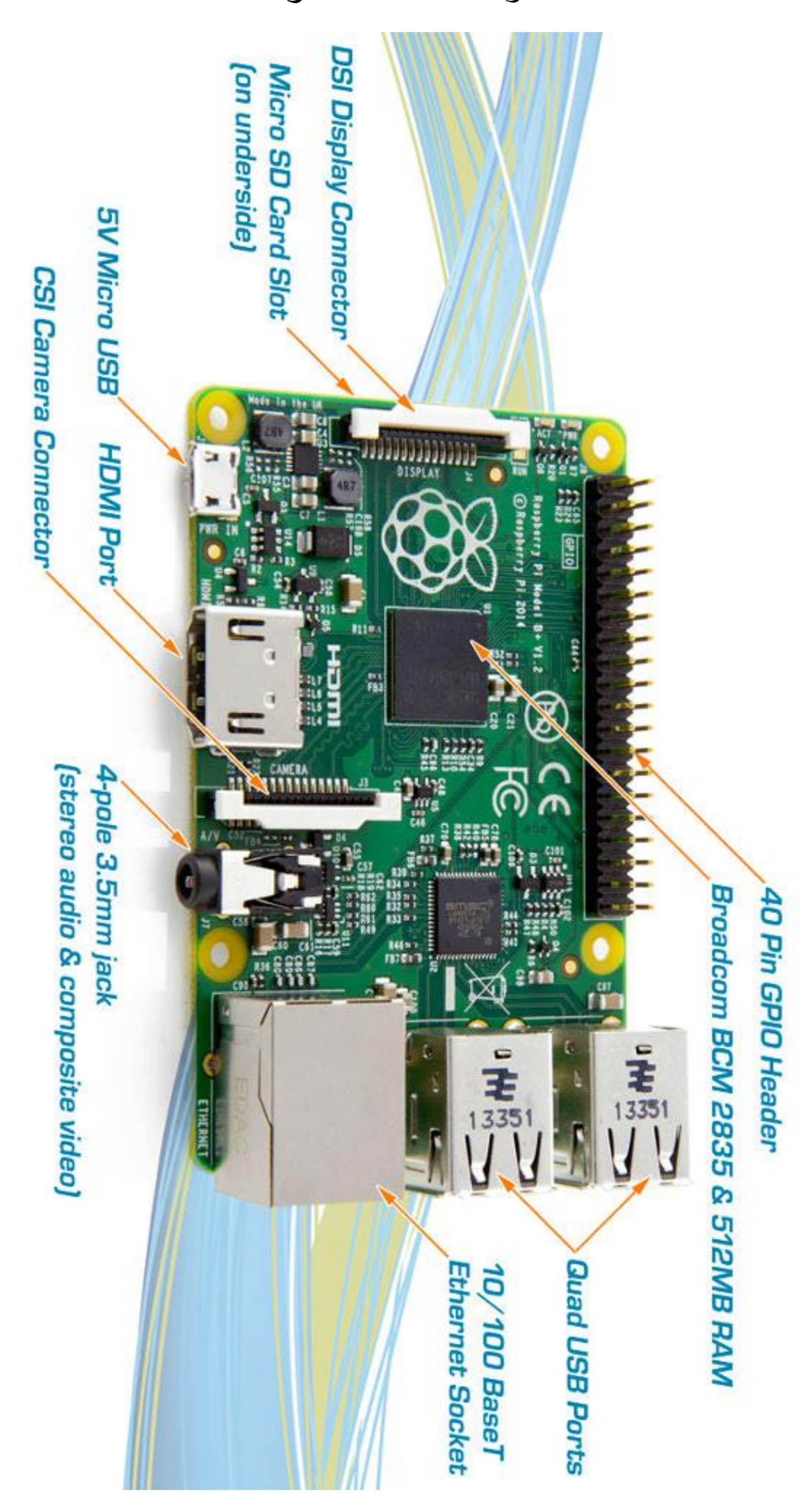
| | |
|---|---|
| **Chip** | Broadcom BCM2836 SoC |
| **Core architecture** | Quad-core ARM Cortex-A7 |
| **CPU** | 900 MHz |
| **GPU** | Dual Core VideoCore IV® Multimedia Co-Processor<br>Provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30 H.264 high-profile decode<br>Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure |
| **Memory** | 1GB LPDDR2 |
| **Operating System** | Boots from Micro SD card, running a version of the Linux operating system |
| **Dimensions** | 85 x 56 x 17mm |
| **Power** | Micro USB socket 5V, 2A |

***

**Connectors:**

| | |
|---|---|
| **Ethernet** | 10/100 BaseT Ethernet socket |
| **Video Output** | HDMI (rev 1.3 & 1.4) |
| **Audio Output** | 3.5mm jack, HDMI |
| **USB** | 4 x USB 2.0 Connector |
| **GPIO Connector** | 40-pin 2.54 mm (100 mil) expansion header: 2x20 strip<br>Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines |
| **Camera Connector** | 15-pin MIPI Camera Serial Interface (CSI-2) |
| **JTAG** | Not populated |
| **Display Connector** | Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane |
| **Memory Card Slot** | Micro SDIO |

مكونات هذا الاصدار :

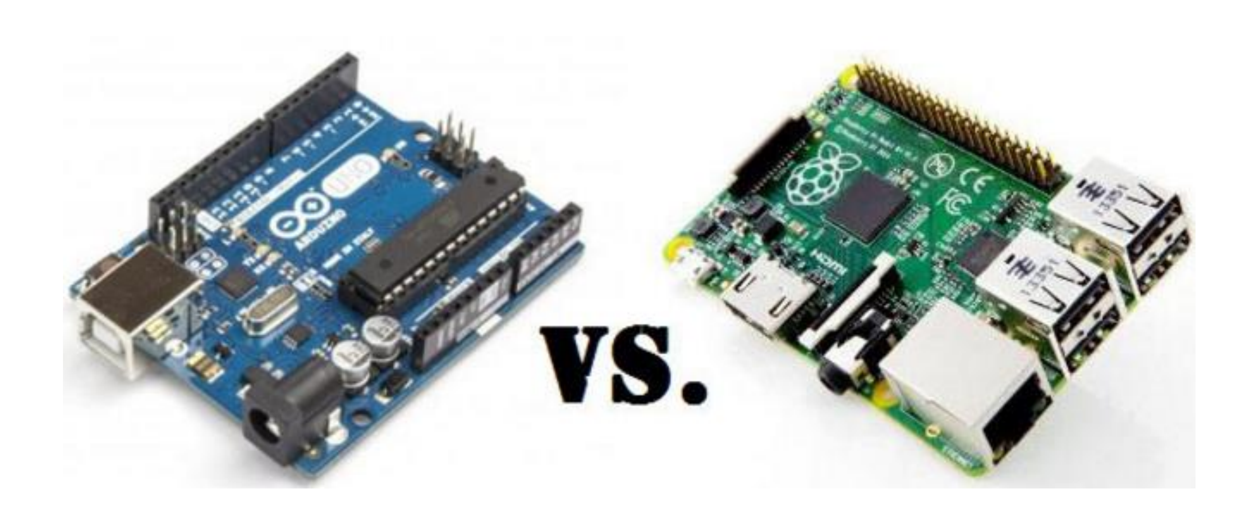


وظائف الاطراف تسمى ب GPIO :

## Raspberry Pi 2 Model B

| GPIO# | NAME | Pin | Pin | NAME | GPIO# |
|---|---|---|---|---|---|
| | 3.3 VDC Power | 1 | 2 | 5.0 VDC Power | |
| 8 | GPIO 8 SDA1 (I2C) | 3 | 4 | 5.0 VDC Power | |
| 9 | GPIO 9 SCL1 (I2C) | 5 | 6 | Ground | |
| 7 | GPIO 7 GPCLK0 | 7 | 8 | GPIO 15 TxD (UART) | 15 |
| | Ground | 9 | 10 | GPIO 16 RxD (UART) | 16 |
| 0 | GPIO 0 | 11 | 12 | GPIO 1 PCM_CLK/PWM0 | 1 |
| 2 | GPIO 2 | 13 | 14 | Ground | |
| 3 | GPIO 3 | 15 | 16 | GPIO 4 | 4 |
| | 3.3 VDC Power | 17 | 18 | GPIO 5 | 5 |
| 12 | GPIO 12 MOSI (SPI) | 19 | 20 | Ground | |
| 13 | GPIO 13 MISO (SPI) | 21 | 22 | GPIO 6 | 6 |
| 14 | GPIO 14 SCLK (SPI) | 23 | 24 | GPIO 10 CE0 (SPI) | 10 |
| | Ground | 25 | 26 | GPIO 11 CE1 (SPI) | 11 |
| 30 | SDA0 (I2C ID EEPROM) | 27 | 28 | SCL0 (I2C ID EEPROM) | 31 |
| 21 | GPIO 21 GPCLK1 | 29 | 30 | Ground | |
| 22 | GPIO 22 GPCLK2 | 31 | 32 | GPIO 26 PWM0 | 26 |
| 23 | GPIO 23 PWM1 | 33 | 34 | Ground | |
| 24 | GPIO 24 PCM_FS/PWM1 | 35 | 36 | GPIO 27 | 27 |
| 25 | GPIO 25 | 37 | 38 | GPIO 28 PCM_DIN | 28 |
| | Ground | 39 | 40 | GPIO 29 PCM_DOUT | 29 |

## *اردوينو ام راسبيري باي ؟*




حسنا بالتاكيد تسائلت مع نفسك هذا السؤال ! وانت تقرأ هذا السؤال (اردوينو ام راسبيري باي؟) يجب ان يرتعش جسدك لشدة المنافسة بين هاذين المتحكمين .

عندما يتعلق الأمر باختيار جهاز كمبيوتر أحادي اللوحة ، فإن Arduino و Raspberry Pi هما من الأسماء الكبيرة التي ستدرسها. ولكن أي واحد يجب أن تختار؟ ما هو أفضل استخدام لاردوينو ؟ ما هي عيوب استخدام Raspberry Pi؟ وكيف تقرر بين الاثنين؟ يمكن أن يكون قرارًا صعبًا ، لذا سنقوم بذكر ببعض التفاصيل هنا من أجلك.

سأناقش Arduino Uno R3 و Raspberry Pi 2 Model B. هناك العديد من إصدارات كل منهما ، وهناك الكثير من البدائل لـ Pi و Arduino التي توفر مواصفات وقدرات مختلفة ، لكن هذين هما الاكثر شيوعا.

وعلى الرغم من أن كلا من آردوينو وراسبري بي هما جهازان صغيران للغاية ، إلا أنهما لديهما أشياء محددة يجيدانها.

اردوينو ، هو متحكم دقيق microcontroller، مما يعني أنه يتفوق في التحكم في الأجهزة الصغيرة مثل أجهزة الاستشعار ، والمحركات ، والأضواء.هذا هو السبب في أفضل استخدام اردوينو لمشاريع مثل بناء ضوء الاستيقاظ ، إنذار للكشف عن الحركة ، أو حتى روبوت صغير.

ستسمع أيضًا أشخاصًا يتحدثون عن "النماذج الأولية"prototyping مع Arduino ، وهي عملية إنشاء جهاز إلكتروني سريع بشكل أولي. إذا كان النموذج الأولي ناجحًا ويعمل الجهاز ، فيمكن صنعه على نطاق أكبر باستخدام لوحات الدوائر المطبوعة.

من ناحية أخرى ، لا يمثل Raspberry Pi متحكمًا دقيق microcontroller، ولا يتم التحكم فيه في أجهزة الاستشعار فقط وأشياء أخرى كهذه. إنه جهاز كمبيوتر بالكامل مزود بنظام التشغيل الخاص به ، ويهدف إلى استخدامه كجهاز واحد.

نظام التشغيل ليس بسهولة الاردوينو ، لذا ستحتاج إلى بعض المعرفة البرمجية للحصول على أقصى استفادة منه ، ولكن هذه إحدى الأشياء التي يقدمها Raspberry Pi الرائعة: مساعدة الأشخاص على تعلم كيفية البرمجة. كما أنه جيد أيضًا في العمل كخادم: يمكنه التواصل مع أجهزة كمبيوتر أخرى ، ويكون بمثابة بديل لجهاز Chromecast ويقدم معلومات ، ويسجل البيانات.

راسبيري هو أفضل في التحدث مع الناس (تشغيل خادم الويب). "اردوينو" أفضل في التحكم او التحدث مع الأجزاء الآلية (المحركات المتحركة) .

عندما تنظر إلى Arduino بجانب Raspberry Pi ، من الواضح جدًا أن الأجهزة تختلف قليلاً بين الاثنين. دعونا نوضح اكثر :

امدادات الطاقة

متطلبات الإمداد بالطاقة في أردوينو بسيطة للغاية ؛ يمكنك توصيله بالكمبيوتر الخاص بك أو حزمة البطارية ، وسيبدأ تشغيل التعليمات البرمجية على الفور.

إذا كانت الطاقة غير متصلة ، فستتوقف. ليست هناك حاجة لتشغيل عملية إيقاف التشغيل. من ناحية أخرى ، فإن Raspberry Pi ، نظرًا لوجود نظام حوسبة أكثر تميزًا في مكانه ، يجب إيقاف تشغيله مثل الكمبيوتر العادي ، ويمكن أن يتضرر بسبب انقطاع التيار الكهربائي.

يتمتع كل من Arduino و Raspberry Pi بسحب طاقة منخفض جدًا ، ويمكن تشغيلهما لمدة طويلة بدون استخدام الكثير من الكهرباء.

## الاتصال

يأتي Raspberry Pi جاهزًا للاتصال بالإنترنت ؛ فهي تحتوي على منفذ إيثرنت مدمج ، ومن السهل جدًا الحصول على وصلة USB USB لتوفير اتصال لاسلكي كذلك (يمكنك رؤية وصلة صغيرة جدًا في الصورة أدناه.




هذا هو أحد الأسباب التي تجعل Pi هو الجهاز المفضل لأشياء مثل خوادم الويب الشخصية وخوادم الطابعة والشبكات VPN

من ناحية أخرى ، لا يمتلك Arduino أي قدرة مضمنة للاتصال.

إذا كنت تريد توصيله بالإنترنت ، فستحتاج إلى إضافة جهاز إضافي مثل المشغل او الغطاء يتضمن منفذ إيثرنت.

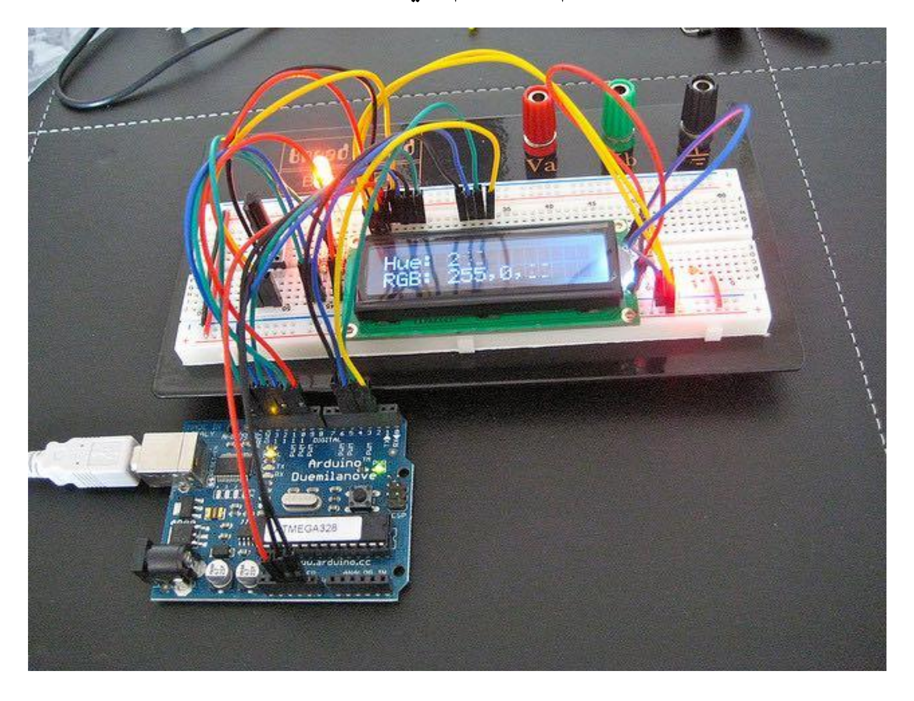
إذا كنت تريد الاتصال بشبكة wifi ، فستحتاج إلى جهاز آخر مرة أخرى. نظرًا لأن Arduino مقصودًا بمشاريع الأجهزة بدلاً من البرامج ، فإنه يحتاج إلى القليل من الترقية لتوصيله.

المداخل والمخارج I/O Pins

دبابيس الإدخال / الإخراج هي التي تسمح للمتحكم أحادي اللوحة بالتحكم في الأشياء المرتبطة به.

على سبيل المثال ، يمكن لجهاز Raspberry Pi إضاءة مصباح LED أو يمكن لاردوينو الخاص بك تنشيط المحرك. إذا كنت تبحث عن توصيلات الأجهزة ، فهذه الدبابيس هي ما تحتاجه.

يحتوي Raspberry Pi 2 على 40 من هذه الدبابيس ، في حين يقدم Arduino Uno 20 ؛ يمكنك رؤية عدد منهم يستخدم في الصورة أدناه:



فرق آخر مهم في دبابيس الإدخال / الإخراج هو الدقة الزمنية التي يمكنك التحكم بها.

نظرًا لأن جهاز Raspberry Pi عبارة عن جهاز كمبيوتر كامل ، فإنه يحتوي على عدد من الأشياء التي تتنافس على الوقت في وحدة المعالجة المركزية (CPU) ، مما يعني أنه قد يواجه بعض الصعوبة في الحصول على توقيت إلى أجزاء صغيرة من الثانية.

ويحتاج البرنامج إلى واجهة ربط بشكل صحيح مع أجهزة الاستشعار والأجهزة الأخرى. من ناحية أخرى ، يمكن أن يغير Arduino الإخراج ويراقب الإدخال على دبابيسه إلى مقدار ضئيل جدًا من الوقت.

## التخزين Storage

يأتي اردوينو مع 32 كيلوبايت من التخزين الداخلي ، وهو ما يكفي لتخزين الكود الذي يقدم التعليمات للبرنامج.

لا يمكنك استخدام هذا السعة التخزينية للتطبيقات أو مقاطع الفيديو أو الصور أو أي شيء آخر. من ناحية أخرى ، لا يأتي جهاز Raspberry Pi مزودًا بأي وحدة تخزين داخلية ، ولكنه يحتوي على منفذ micro SD ، لذا يمكنك إضافة قدر التخزين الذي تريده. إن إضافة سعة تخزين تبلغ 32 غيغابايت لن تكلفك سوى 12 دولارًا فقط باستخدام بطاقة Micro SD ، ويمكنك بسهولة إضافة ما يصل إلى 128 أو 256 غيغابايت إذا كنت بحاجة إليها.

## USB

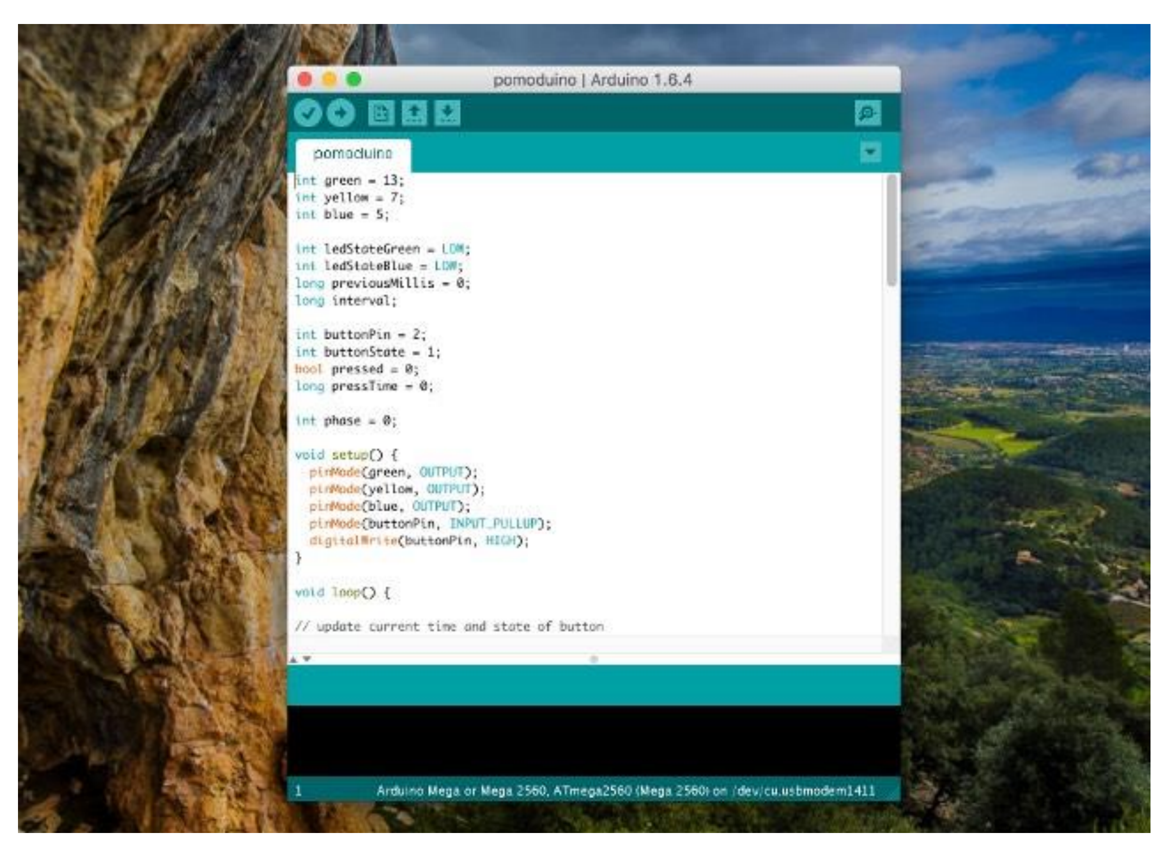
نظرًا لأن نظام اردوينو لا يهدف بشكل اساسي إلى التواصل مع أجهزة الكمبيوتر ، فإنه لا يأتي قياسيًا مع أي منافذ USB التي يمكنك استخدامها لهذا النوع من الاتصالات الكبيرة. يمكن استخدام منفذ واحد لتوصيل Arduino بالكمبيوتر عبر منفذ USB للكمبيوتر الخاص بك ، ولكن من ناحية أخرى ، فإن جهاز Raspberry Pi يحتوي

على أربعة منافذ USB التي يمكنك استخدامها لتوصيله بجهاز توجيه أو طابعة أو محرك أقراص ثابت خارجي أو مجموعة متنوعة من الأجهزة الأخرى.

## البرمجيات Software

الآن بعد أن وضعنا الاختلافات بين أجهزة Arduino و Raspberry Pi ، يمكننا التحدث عن البرامج. لفهم حقًا متى ترغب في استخدام لوحة واحدة أو أخرى ، ستحتاج إلى معرفة ما يمكن لكل واحد القيام به ، ويعتمد الكثير من ذلك على البرنامج.

اردوينو  لديه إمكانات أساسية جدًا للبرمجة التي يتلقاها لتغيير وظائف الجهاز الذي يتصل به ، ولكن لا يوجد في اللوحة نظام تشغيل أو أي نوع من الواجهة إلى جانب بيئة التطوير المتكاملة لاردوينو (IDE).



باستخدام IDE ، ستقوم بإنشاء مجموعة من الأوامر التي سيقوم Arduino بتفسيرها وتفعيلها.

يمكن لمجموعة بسيطة من التعليمات أن تقول شيئًا مثل "تشغيل الضوء الأحمر لمدة ثلاث ثوانٍ ، إيقاف تشغيله ، تشغيل الضوء الأخضر لمدة ثلاث ثوانٍ ، إيقاف تشغيله ، تكرار. من الواضح أنه يمكنك القيام بأشياء أكثر تعقيدًا ، ولكن ستظلّ بحاجة إلى إنشاء البرنامج بنفسك.

لحسن الحظ ، يوجد مجتمع اردوينو ضخم يمتد عبر العالم بأكمله ، مما يعني أنه إذا كان هناك شيء تريد القيام به باستخدام Arduino ، فمن المحتمل أن يكون شخص ما قد فعل ذلك.

يمكنك إلقاء نظرة على الكود الخاص بهم ، وتعديله ، وجعل ما يفعله اردوينو بالضبط ما تريده. هذا هو وسيلة رائعة لتعلم مبادئ البرمجة والنماذج الجاهزة ، كذلك ، وهذا هو السبب في اردوينو هو خيار عظيم لأي شخص مهتم بالإلكترونيات.

في المقابل ، تأتي Raspberry Pi مزودة بنظام تشغيل وظيفي بالكامل يدعى Raspbian. يعتمد نظام التشغيل هذا على Debian Linux ، وتم إنشاؤه خصيصًا لـ Pi. هناك عدد من أنظمة التشغيل الأخرى التي يمكنك استخدامها مع اللوحة ، ومعظمها يستند إلى Linux ، ولكن يمكن أيضًا تثبيت Android.

ليست أنظمة التشغيل هي البرامج الوحيدة التي يديرها Pi ؛ هناك أيضًا عدد من التطبيقات المفيدة التي يمكنك استخدامها لإنجاز مهام مختلفة.

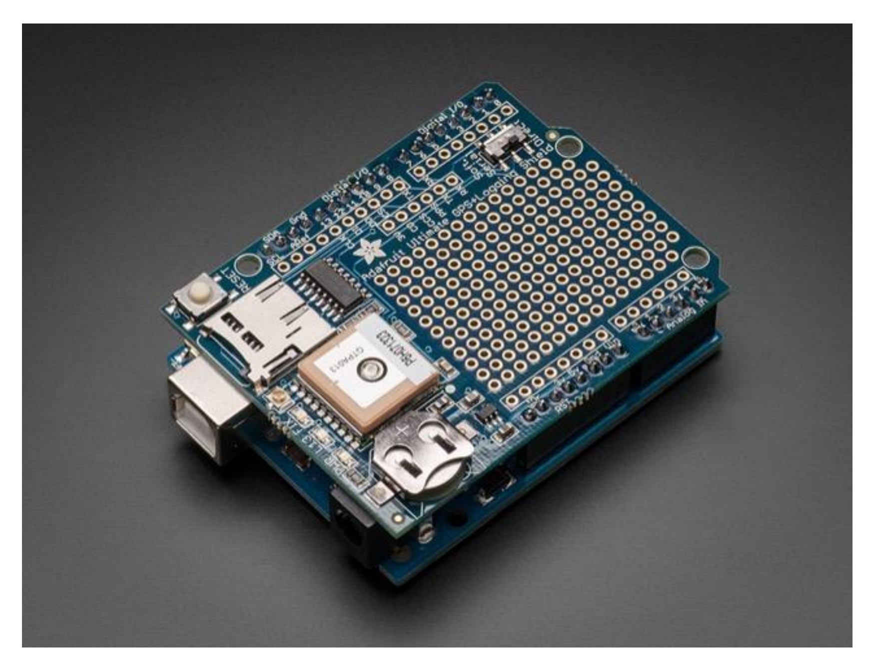
أحد الاستخدامات الأكثر شيوعًا لـ Raspberry Pi هو كخادم وسائط ، حيث يعتبر كل من Kodi و Plex من التطبيقات الشهيرة. يمكنك تنزيل الألعاب وتطبيقات الخوادم والآلات الحاسبة وحتى مجموعة ال Office.

بالطبع ، يمكنك كتابة برامجك الخاصة لـ Raspberry Pi ، وهذا هو أحد أفضل الأسباب للحصول على متعة تعلم البرمجة.

بايثون Python هي اللغة الموصى بها للـ Pi ، لكن C و C ++ و Java و Ruby جميعها مثبتة مسبقًا على اللوحة. بينما يمكن تعديل Arduino لدعم لغات أخرى ، لكن لغة Arduino الأصلية هي الخيار الأفضل. إذا كنت تتطلع إلى تعلم لغة أكثر فائدة ، فستقدم لك Pi المزيد من الخيارات.

إن كلا من Arduino و Raspberry Pi هما آلات صغيرة قادرة على مساعدتك على التعلم و القيام بالكثير من الأشياء ، ولكن في مرحلة ما ، ربما ترغب في تجاوز الأساسيات وتجربة شيء أكثر تقدمًا.

هذا هو واحد من الاماكن التي يلمع Arduino. فيها هناك المئات من الرقائق التي تسمح لك بتوسيع إمكاناتك مع الاشياء مثل إيثرنت واتصال واي فاي ، تحكم أفضل في المحركات ، قدرات مكبر الصوت والميكروفون ، شاشة تعمل باللمس ، كاميرات ، أجهزة إرسال لاسلكية ، معالجة رسومات ، وأي شيء آخر يمكنك التفكير فيه من. من 10 إلى 40 دولارًا ، يمكنك تحويل Arduino إلى شيء آخر تمامًا (مثل غطاء Adafruit GPS).



تسمى هذه الرقاقات الاغطية والتي سبق درسناها ، ويتم تثبيتها بسهولة. كل ما عليك فعله هو وضعهم على رأس Arduino و- في بعض الحالات - لحامها في مكانها. مما يجعل التثبيت محكما .

هناك عدد من الاغطية المتاحة التي تضيف أجهزة إضافية إلى Pi ، ومع ذلك ، تعطيك بعض الخصائص المثيرة للاهتمام.

على سبيل المثال ، يمكنك إضافة أجهزة استشعار سعوية ونظام تحديد المواقع العالمي وشاشة تعمل باللمس ولوحات RGB وحتى مستشعر حركة ثلاثي الأبعاد.

ومع ذلك ، حتى مع وجود هذه الخيارات ، فإن Raspberry Pi ليس لديه الكثير من الخيارات لإضافة وظائف. لا يعني أن Pi غير قادر ؛ لا يزال بإمكانك فعل أي شيء تريده تقريبًا ، قد تحتاج فقط إلى المزيد من الإبداع (أو إرفاقه بـ Arduino!).

كيف تقرر بين اردوينو وراسبري بي؟

الآن بعد أن رأيت بالضبط كيف يختلف اردوينو وراسبري بي ، يجب أن تكون لديك فكرة جيدة حول كيفية اتخاذ قرار بين الاثنين إذا كنت ترغب في الحصول على واحدة.

اذا كنت ترغب في إنشاء أجهزة ، مثل الروبوتات ، وأجهزة ضبط الوقت ، وأجهزة الاستشعار ، فإن Arduino هو الطريق الذي يجب أن تسلكه ؛ واجهة البرنامج منخفضة المستوى للتعلم واتصالات I / O سهلة تجعلها أفضل طريقة للبدء إذا كنت ترغب في بناء شيء ما.

من ناحية أخرى ، فإن جهاز Raspberry Pi يقوم بإنشاء تطبيقات رائعة مثل نظام تخزين بيانات ، وهو رائع لتعلم البرمجة باللغات التقليدية. إذا كنت ترغب في التواصل مع أجهزة كمبيوتر أخرى ، فإن Pi هو عالمك .



لكن لماذا تحد نفسك؟ لماذا لا تحصل على كليهما؟ إن كلاهما بأسعار معقولة جدًا ، ويمكنك الحصول على مجموعات بداية أقل من 100 دولار والتي تشمل كل ما تحتاجه لبدء العمل في المشاريع. من روبوت بسيط إلى خادم ويب كامل ، يمكنك الحصول على مجموعة بسيطة لمساعدتك في عملية الإعداد.

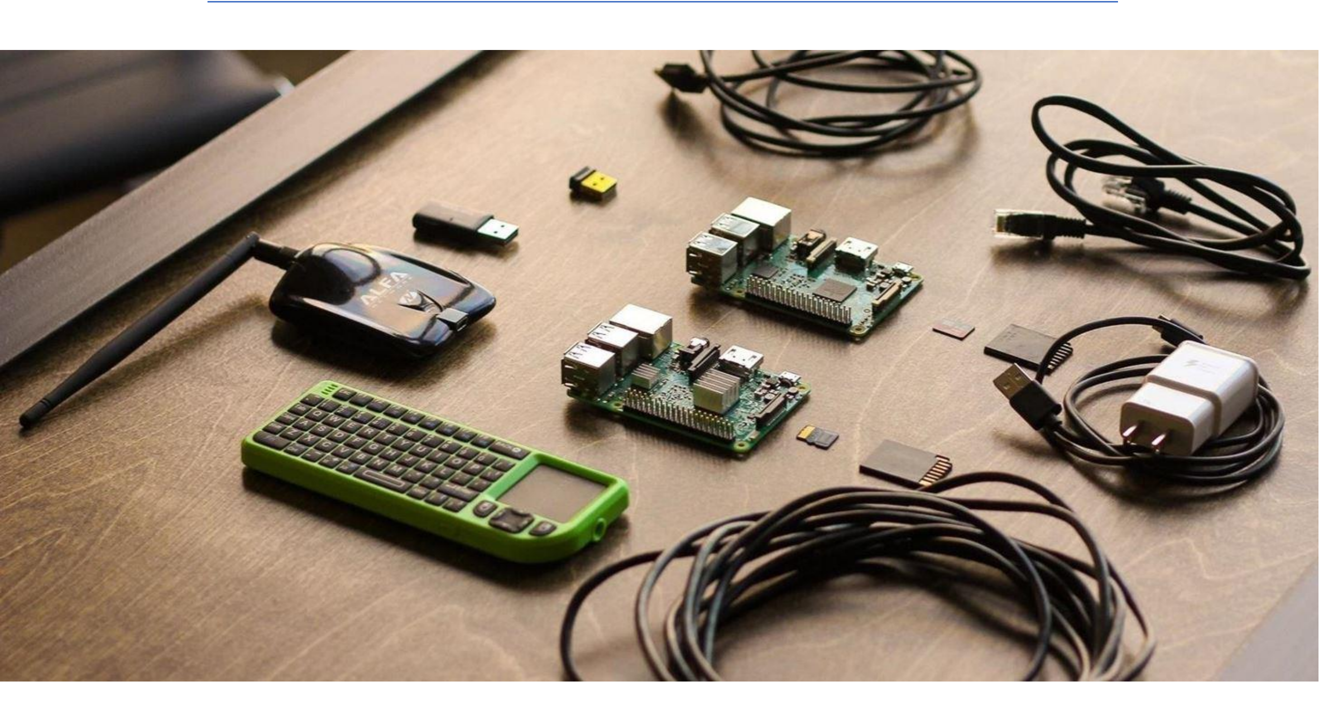
وعندما تبدأ في التعلم وتصبح أكثر تقدمًا ، يمكنك استخدام كل من Arduino و Pi معًا لتشغيل المستشعرات و servos مع إرشادات ودروس عبر الكتب الإنترنت ! اذا الخيارات لا حدود لها.

جدول المقارنة

| | Arduino Uno | Raspberry Pi 2 Model B |
|---|---|---|
| **Cost (base model)** | 20 | 39 |
| **Processor** | 16MHz AVR ATmega328P | 900 MHz Broadcom ARM Cortex-A7 |
| **Storage** | 32 KB | n/a |
| **RAM** | 2 KB | 1 GB |
| **I/O pins** | 20 | 40 |
| **OS** | n/a | Raspbian, other varieties of Linux, Android |
| **Languages** | Arduino, | Python, C, C++, Java, Ruby |
| **Best for** | Hardware / prototyping | Software / server |
| **Power supply** | 5V USB or DC jack | 5V USB |

قم بالاطلاع على هذه المعلومات : **https://elinux.org/RPi_BCM2835_GPIOs**

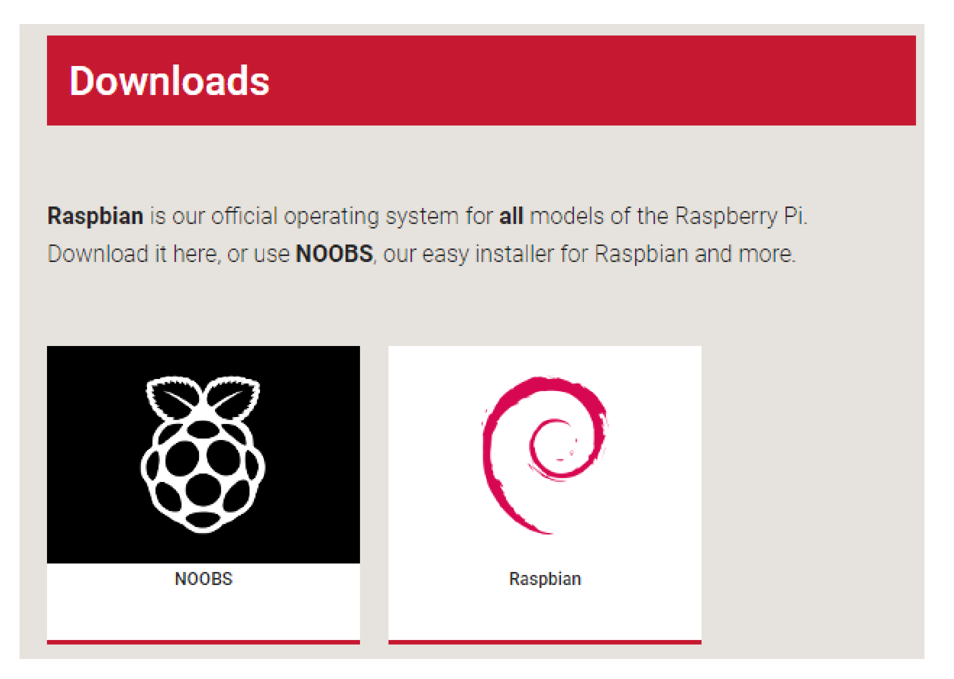
# تهيئة الراسبيري باي



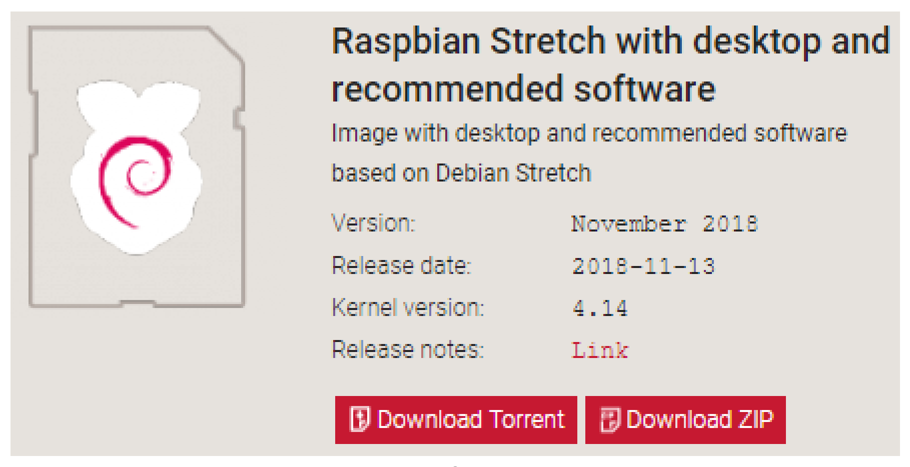

في هذا الدرس سوف نتعلم كيف نبدء مع الراسبيري باي , بعد شراء اللوح الالكتروني لا تستطيع البدء مباشرة في تجهيز بيئة العمل وبرمجتها للتحكم في مشاريعك الخاصة فيوجد خطوات لا بد من عملها قبل البدء وهي ماسوف نتطرق له في هذا الدرس , يحتوي هذا الدرس على جزئين , الجزء الاول يتعلق بتنصيب نظام التشغيل والجزء الثاني يتعلق في اعدادات العتاد الالكتروني , اي الادوات المطلوبة لبدء العمل .

تحدثنا سابقا ان الراسبيري باي تعمل على انظمة التشغيل لينكس , من احدى التوزيعات الاكثر شهرة والتي تعتبر التوزيعة الرسمية للراسبيري باي هي توزيعة النظام راسبيان Raspbian هي توزيعة مفتوحة المصدر ومجانية , ولتحميل هذا النظام نذهب الى الرابط التالي , او من محتويات قرص الكتاب :

/www.raspberrypi.org/downloads




ثم نقوم باختيار Raspbian ونحمل الملف التالي بالضغط على Download ZIP



عند اكتمال التحميل سيظهر الملف بالشكل المضغوط , يجب فك الضغط باستخدام برنامج ال WIN ZIP او WIN RAR:




الان بعد اكتمال تنزيل النظام بصيغة img. تبقى لنا استخدام SD-Card لتنصيب النظام عليها حتى يتم ادخالها الى الراسبيري باي .

نحتاج الى SD-Cardبسعة يختارها المستخدم كحد ادنى 32 جيجا (موصى بها)



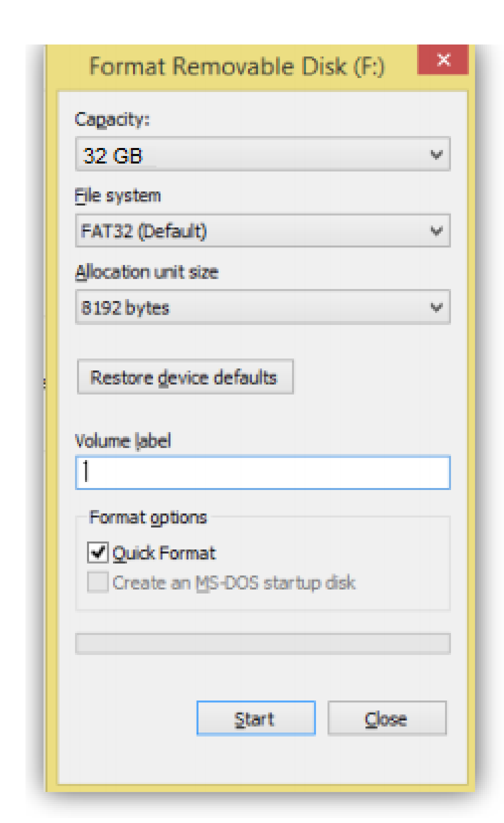

الان نقوم بتوصيل ال SD مع قارئة الذواكر التالية :

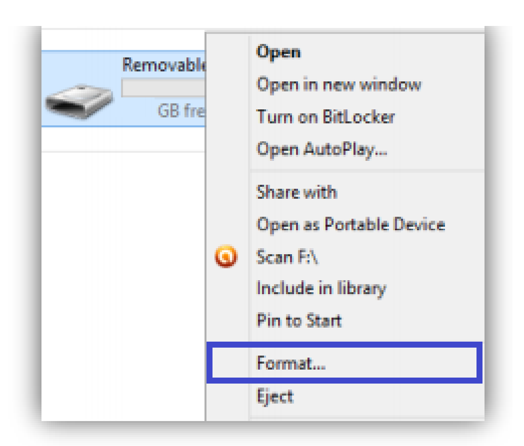


الان لا يمكن عمل نسخ copy ولصق paste للنظام الذي تم تحميله يجب استخدام برنامج خاص (Win32 Disk Imager) لتنصيب النظام على ال SD-Card يتم تحميل البرنامج من الرابط ادناه او من محتويات ملفات قرص الكتاب :

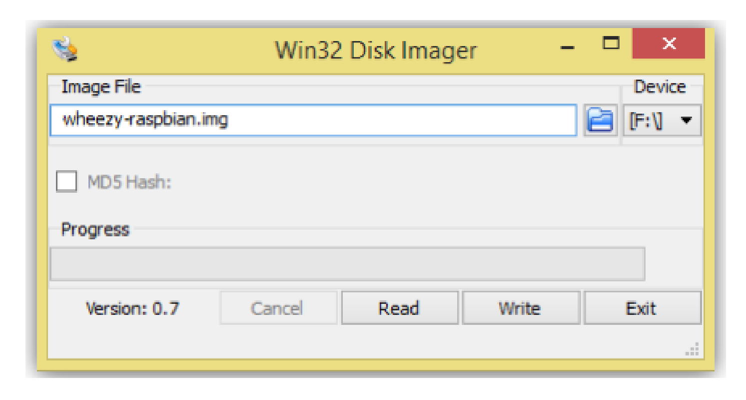
https://sourceforge.net/projects/win32diskimager/files/latest/download

نقوم بادخال ال SD-Card في القارئة لادخالها في جهاز الحاسوب ثم نعمل formatلل SD-Card

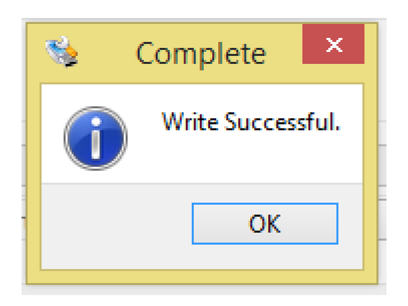



الان نفتح برنامج ال(Win32 Disk Imager)



نختار ملف النظام ثم نضغط على Write ننتظر قليلا وسيبدأ تحميل النظام الى ال SD-Card وعندما ينتهي تتظهر الرسالة التالية :




حسنا الان لقد انتهينا من رفع البرنامج من جهاز الحاسوب الى ال SD-Card وحتى نبدأ في الراسبيري باي يلزمنا بعض الادوات وهي ما يلي :

مصدر طاقة قد يكون شاحن الهاتف مخرج الراسبيري باي هو micro USB ويحتاج الى 5 فولت و 700 ملي امبير على الاقل .



ونحتاج ايضا الى لوحة مفاتيح وفأرة ووصلة HDMI او كيبل التلفاز RCA



قم بادخال ال SD-Card بعد توصيل الاسلاك بالاضافة الى كيبل الانترنت للحصول على خدمة الانترنت داخل النظام و سيصبح الشكل كالتالي :

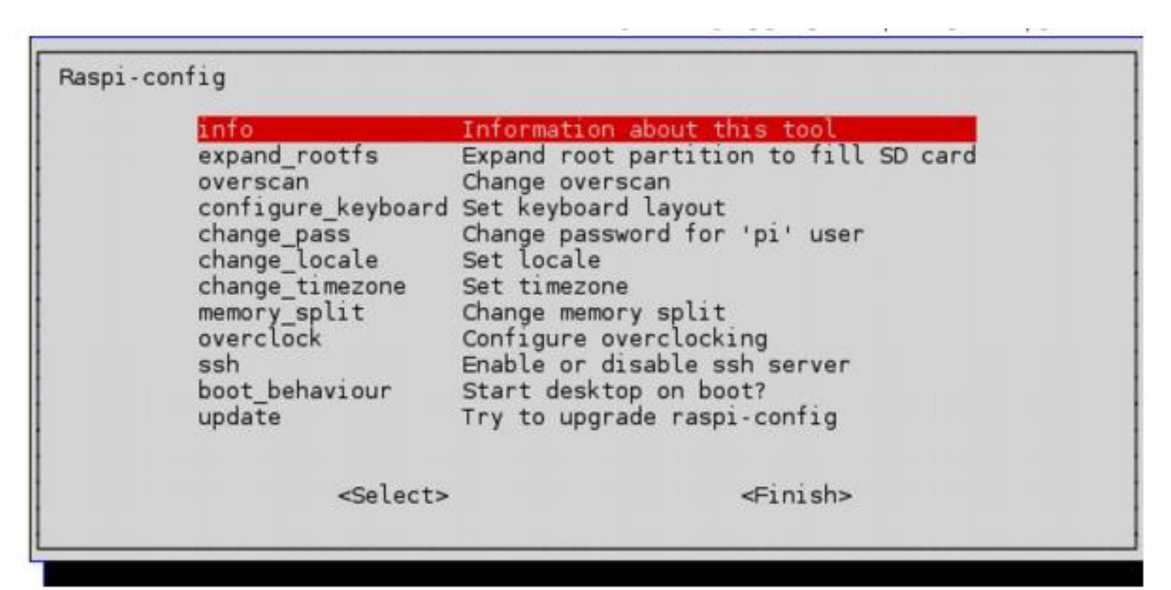


انصح باستخدام الغطاء الخاص بالراسبيري لانها للوحة حساسة للظروق الخارجية ويوجد عدة اشكال وانواع منها :






بعد التاكد من ان كل التوصيلات صحيحة , يمكنك اخيرا تشغيل الطاقة في الراسبيري باي , وعند تشغيل النظام لاول مرة ستظهر الشاشة التالية على شاشة التلفاز او الحاسوب :




Expand rootfs: هذا الخيار يقوم بتوسيع نظام الملفات بحيث يستخدم مساحة بطاقة الذاكرة كاملةً (افتراضياً نظام لينكس يستغل حوالي ٢ جيجا فقط من المساحة)

Overscan: إذا كان لديك شاشة عريضة أو شاشة HD ففي الغالب ستجد أن أجزاء من النصوص والأشياء المعروضة تذهب إلى جوانب الشاشة و لا تظهر بصورة صحيحة، هذا الخيار يحل هذه المشكلة ان وجدت.

Configure Keyboard: ضبط لغة الكتابة مع العلم أن اللغة الافتراضية هي الإنجليزية (البريطانية) يمكنك هذا الخيار من إضافة المزيد من اللغات (مثل أضافة العربية)

Change Pass: يمكنك من تغير كلمة السر الأساسية لدخول الجهاز مع العلم أن الكلمة الافتراضية هي raspberry و أسم المستخدم الافتراضي هو pi

Change Locale: الخيار المسؤول عن ضبط البلد، افتراضيا ستكون البلد (بريطانيا) يمكنك تغييره إلى أي بلد تشاء

Change time zone: كسابقه ولكن هذا للتوقيت المحلي

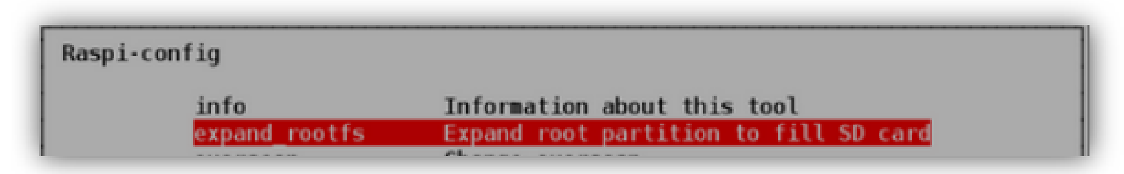
Memory split: هذا الخيار يمكنك من التحكم في مقدار الذاكرة التي يستخدمها المعالج و معالج الرسوميات (أنصحك بأن لا تعبث مع هذا الخيار)

Overclock: خيار التحكم في سرعة المعالج، كما تعلمنا سابقاً ان سرعة المعالج الافتراضية هي ٧٠٠ ميجا هرتز ولكن يمكنك مع هذا الخيار أن تسرع المعالج حتى ٩٠٠ أو ١٠٠٠ ميجا هرتز

تحذير: عمل كسر للسرعة يحتاج إلى تبريد عالي وإجراءات خاصة، لا تعبث مع هذا الخيار الأن لأنه قد يؤدي إلى تلف جهازك.

- SSH: هذا الخيار يقوم بتشغيل خاصية تسمى (Secure Shell server) وهذه الخاصية تسمح لك بالوصول إلى جهازك عن بعد باستخدام شبكة داخلية مثلًا (خاصية مفيدة جدًا سنتحدث عنها بالتفصيل في فصل كامل عن تقنيات التشغيل و التحكم عن بعد)
- Boot Behavior: هذا الخيار يتيح لك الوصول إلى الواجهة الرسومية للنظام أو سطر الأوامر
- Update: هذا الخيار لتحديث جميع البرامج و قائمة الضبط إذا كنت متصلًا بالإنترنت مع العلم ان هذا الخيار لا يقوم بتحديث نظام التشغيل نفسة
- Finish: سوف يقلع الجهاز إلى الواجهة الرسومية للنظام

سنقوم بالدخول الى الخيار Expand rootfs حتى يستطيع نظام لينكس استغلال مساحة بطاقة الذاكرة بالكامل وبدون تنفيذ هذا الخيار لن يرى النظام أكثر من ٢ جيجا فقط من الذاكرة، كل ما عليك فعلة هو أن تضغط على هذا الخيار ثم تنتظر حتى ظهور رسالة تفيد بانتهاء عملية التوسيع ثم اضغط على Finish لعمل إعادة تشغيل والدخول للواجهة الرسومية.



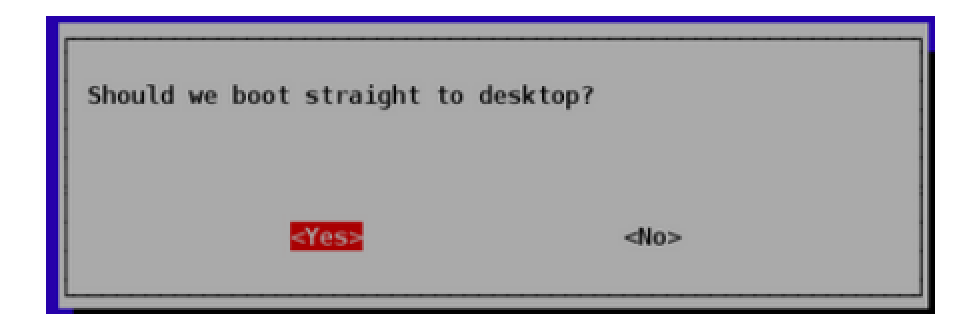



بعد الأنتهاء من توسيع البطاقة سنضبط الراسبيري باي للدخول تلقائيا إلى سطح المكتب و ذلك عبر الدخول إلى الخيار Boot behavior

```
Raspi-config

    info                Information about this tool
    expand_rootfs       Expand root partition to fill SD card
    overscan            Change overscan
    configure_keyboard  Set keyboard layout
    change_pass         Change password for 'pi' user
    change_locale       Set locale
    change_timezone     Set timezone
    memory_split        Change memory split
    ssh                 Enable or disable ssh server
    boot_behaviour      Start desktop on boot?
    update              Try to upgrade raspi-config

            <Select>                    <Finish>
```

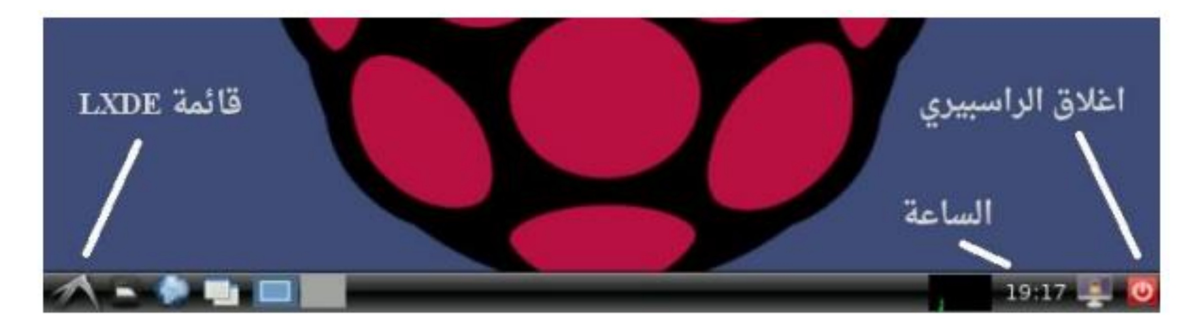


الان بعد الضغط على finish سوف يقلع النظام للمرة الاولى

لاحظ انه يشبه نظام ويندو كثيرا , لكن في هذا الدرس نستهدف الراسبيري باي سوف نقوم بشرح نظام راسبيان كاملا في كتاب منفصل في وقت لاحق .

في هذا الكتاب نريد ان نفتح نافذة الاوامر التي بها نبرمج لوحة الراسبيري باي ونكتب اومر بسيطة للمبتدئين , اين نكتب البرامج الخاصة بالراسبري باي ؟

نضغط على قائمة LXDE ثم accessories ثم نختار البرنامج LXTerminal



وستظهر النافذة التالية :

قبل البدء للتحكم في نقوم بتحميل المكتبة المسؤولة عن التحكم الالكتروني بكتابة الكود التالي داخل شاشة الاوامر :

```
sudo apt-get apt-get update

sudo apt-get apt-get install -y python-dev python-rpi.gpio
```




## Programming for the Raspberry Pi

سوف نقوم في التعرف على للغة بايثون بشكل سريع (بهدف فهم الكودات الابتدائية) في الاصدار الثاني من السلسلة سيتم الشرح التفصيلي للراسبيري باي مثل دروس الاردوينو وملحقاته . ولكن حتى نتمكن من عمل بعض المشاريع الابتدائية باستخدام الراسبيري باي والبايثون نوضح بعض المفاهيم الاساسية للغة.

استخدام البايثون كآلة حاسبة

```
>>> 1 + 2
3
>>> name = "Sarah"
>>> "Hello " + name
'Hello Sarah'
```

Hello world في البايثون

```
print("Hello world")
```

بسيطة اليس كذلك ؟

لا تستخدم Python الأقواس المتعرجة ، بل تتطلب المسافة البادئة على سبيل المثال الدوران في بايثون :

```
for i in range(10):
    print("Hello")
```

المسافة البادئة ضرورية سيكون السطر الثاني الذي له مسافة بادئة هو جزء من الحلقة ، ويكون السطر الثاني غير خارج الحلقة كالتالي :

```
for i in range(2):
    print("A")
    print("B")
```

النتيجة

```
A
B
A
B
```

اما لو كان

```
for i in range(2):
    print("A")
print("B")
```

النتيجة

```
A
A
B
```

معظم لغات البرمجة تستخدم نفس المتغيرات مثل ما يلي :

```
name = "Bob"
age = 15
```

لاحظ أن أنواع البيانات لم يتم تحديدها مع هذه المتغيرات ، حيث يتم الاستدلال على الأنواع لاحقا

```
age = 15
age += 1  # increment age by 1
print(age)
```

يتم تجاهل التعليقات في البرنامج ولكن تستخدم لتتمكن من ترك الملاحظات ، ويتم بواسطة الرمز # هاش. تستخدم التعليقات متعددة الأسطر علامات اقتباس ثلاثية مثل:

```
"""
This is a very simple Python program that prints "Hello".
That's all it does.
"""


print("Hello")
```

بايثون لديها أيضا قوائم (تسمى المصفوفات في بعض اللغات) والتي هي مجموعات من البيانات من أي نوع:

```
numbers = [1, 2, 3]
```

نعرف القوائم باستخدام الأقواس المربعة [] ويتم فصل كل عنصر بفاصلة.

بعض أنواع البيانات قابلة للتكرار ، مما يعني أنه بإمكانك التكرار فوق للقيم التي تحتويها. على سبيل المثال :

هذا يأخذ كل عنصر في numbers من القائمة ويطبعه:

```
numbers = [1, 2, 3]


for number in numbers:

    print(number)
```

النتيجة :

```
1
2
3
```

او

```
dog_name = "BINGO"


for char in dog_name:

    print(char)
```

النتيجة

```
B
I
N
G
O
```

```
n = 0

for i in range(1, 101):
    n += i

print("The sum of the numbers 1 to 100 is:")
print(n)
```

النتيجة

```
5050
```

نوع البيانات الصحيح integer غير متكرر ومحاولة سيؤدي إلى حدوث خطأ. فمثلا:

```
for i in 3:
    print(i)
```

تعطي لنا النتيجة

```
TypeError: 'int' object is not iterable
```

لكنا يمكن تكرارها باستخدام range

```
for i in range(3):
    print(i)
```

اما الشرط فهو كالتالي :

```
name = "Joe"


if len(name) > 3:
    print("Nice name,")
    print(name)
else:
    print("That's a short name,")
    print(name)
```

النتيجة

```
That's a short name,
Joe
```

حسنا لنبدأ في بعض المشاريع وبها نوضح التفاصيل التو نواجهها

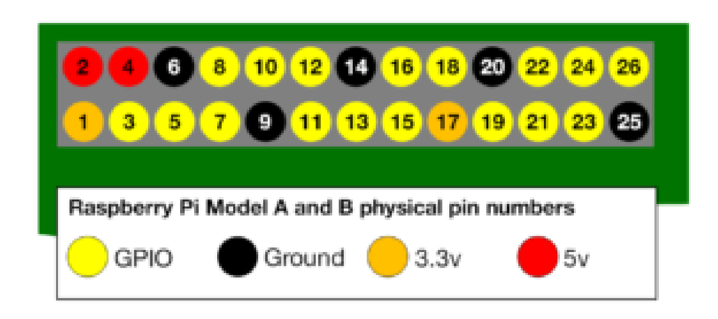
سوف نعمل على تركيب مؤشر LED باللون الأحمر وإضفاء اللمعة باستخدام نص Pythonسيرشدك هذا المثال كيفية إعداد الدارة وتوصيلها إلى Raspberry Pi وكيفية كتابة نص Python النصي الذي يجعل ال LED يومض.

سوف نستخدم Raspberry Pi 3

الخطوة الأولى في هذا المشروع هي تصميم دارة LED بسيطة. ثم سوف نجعل دائرة LEDقابلة للتحكم من Raspberry Pi من خلال توصيل الدائرة إلى دبابيس الإدخال / الإخراج (GPIO) .

قم بتوصيل ال  LED بالراسبيري باي كالتالي :

عند تركيب الدائرة. السلك الطويل هو الجانب الموجب الذي يسمى أيضا الأنود ، والسلك القصير هو الجانب السالب الذي يسمى الكاثود. يجب أن تكون طويلة متصلة مع المقاومة ويجب ربط القصيرة إلى الأرض GND والسلك الاخر الى دبوس 8 على Raspberry Piكما هو موضح في الرسم التخطيطي.



الكود البرمجي :

```
import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library
from time import sleep # Import the sleep function from the time module
GPIO.setwarnings(False) # Ignore warning for now
GPIO.setmode(GPIO.BOARD) # Use physical pin numbering
GPIO.setup(8, GPIO.OUT, initial=GPIO.LOW) # Set pin 8 to be an output pin and set initial value to low (off)
while True: # Run forever
 GPIO.output(8, GPIO.HIGH) # Turn on
 sleep(1) # Sleep for 1 second
 GPIO.output(8, GPIO.LOW) # Turn off
 sleep(1) # Sleep for 1 second
```

وعند تنفيذ البرنامج ستلاحظ وميض المصباح LED

# وحدة التحكم القابلة للرمجة PLC



- ما هو الPLC :

هي اختصارا ل (Programmable Logic Controllers) والتي تعني وحدة التحكم المنطقية القابلة للبرمجة.

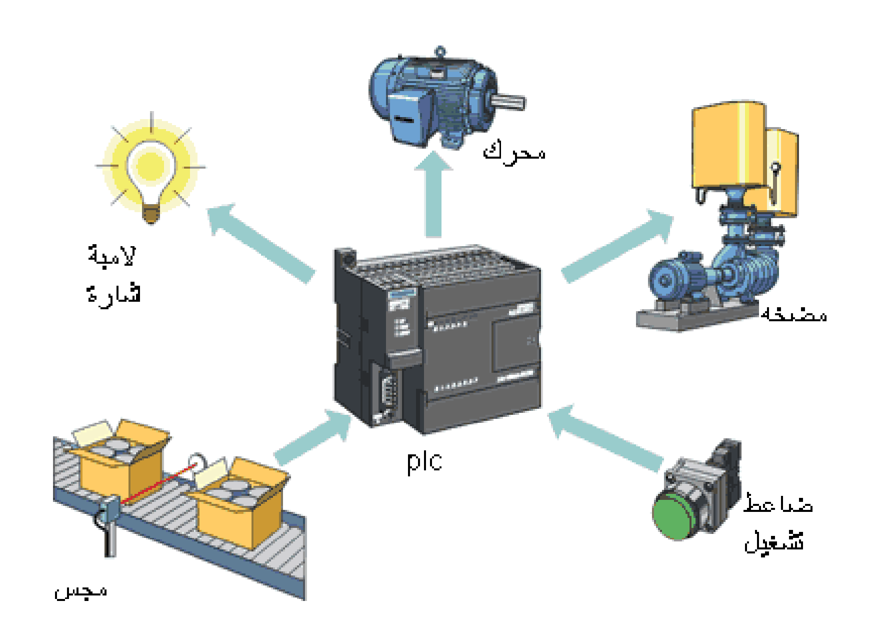
هو حاسوب رقمي يستعمل في أتمتة العمليات الكهروميكانيكية معالج دقيق يستخدم للتحكم في العمليات المختلفة مثل التحكم في الألات والتحكم في العمليات الصناعية المختلفة. يختلف الحاسوب المستخدم في التحكم الصناعي PLC عن الحاسوب الشخصي PC في تركيزه على إدارة عمليات الإدخال والإخراج المنطقي وعمليات القياس والتحكم التماثلي وذلك بفضل وحدات الإدخال والإخراج المختلفة. هذا الحاكم له القدرة على تخزين التعليمات لينفذ وظائف تحكم مثل التوقيت ، العد، معالجة البيانات ، الإزاحة ، الحساب و الإتصال للتحكم في الألات و العمليات الصناعية. في الحاسوب العادي عادة تكون وحدات الإدخال والإخراج مخصصة للغرض الشخصي مثل لوحة

المفاتيح، الفأرة، الشاشة أو الطابعة وغيرها. أما في البي إل سي (PLC) فإن وحدات أخرى هي التي تكون أكثر أهمية وتختلف تماما عن الوحدات السابقة.

في التطبيقات الصناعية الصغيرة توجد وحدة متكاملة شبيهة بجهاز التحكم المنطقي PIC Chip أو PIC microcontroller الذي درسناه سابقا وتستخدم عادة في الأجهزة الإلكترونية لتنفيذ عدد محدود من التعليمات البرمجية.

تختص البرمجه بـ ( PLC ) بالعمليات المنطقية المختلفة (AND , OR , XOR) وعمليات توصيل او فصل المكونات وتصمم الدوائر داخل الجهاز بحيث تتحمل الاهتزازات والحرارة والرطوبة والضوضاء.

تأتي أجهزة PLC بأشكال وبأحجام عديدة وأحيانا ضمن أجهزة صناعية وبعضها متطور يمكن برمجتها مباشرة بواسطة شاشة LCD صغيرة ودون استخدام شاشة كمبيوتر




- مميزات الـ PLC

1 - يمكن تعديل البرنامج بسهولة وبدون اللجوء الى تغيير التوصيلات سواء فى الدخل او الخرج وبالتالى تكون النتيجة نظام مرن يمكن استخدامه فى وظائف التحكم ذات الطبيعة المتغيرة

2 - انخفاض التكلفة حيث انه بادخال التقنيات الحديثة امكن تخفيض تكلفة الـ PLC بحيث اصبح فى كثير من الحالات اوفر من الطرق التقليدية للتحكم

3 - يمكن اجراء تنفيذ للبرنامج قبل تركيب وحدة الـ PLC على المعدة الصناعية

4 - سهولة وسرعة التنفيذ للبرنامج في عمل التغيرات لنظام التحكم

5 - الحماية من العبث فى البرنامج وسهولة التعامل

6 - تصحيح الاخطاء وتحديد مكان الاخطاء حيث يمكن تحديد مكان الاخطاء بدقة وبالتالى يمكن اجراء تعديل فى البرنامج بحيث يستمر اداء الماكينة طبيعى حتى يتم اصلاح العطل

7 - صغر حجمه مما يؤدى الى سهولة التحكم فى وضعه فى المكان المناسب

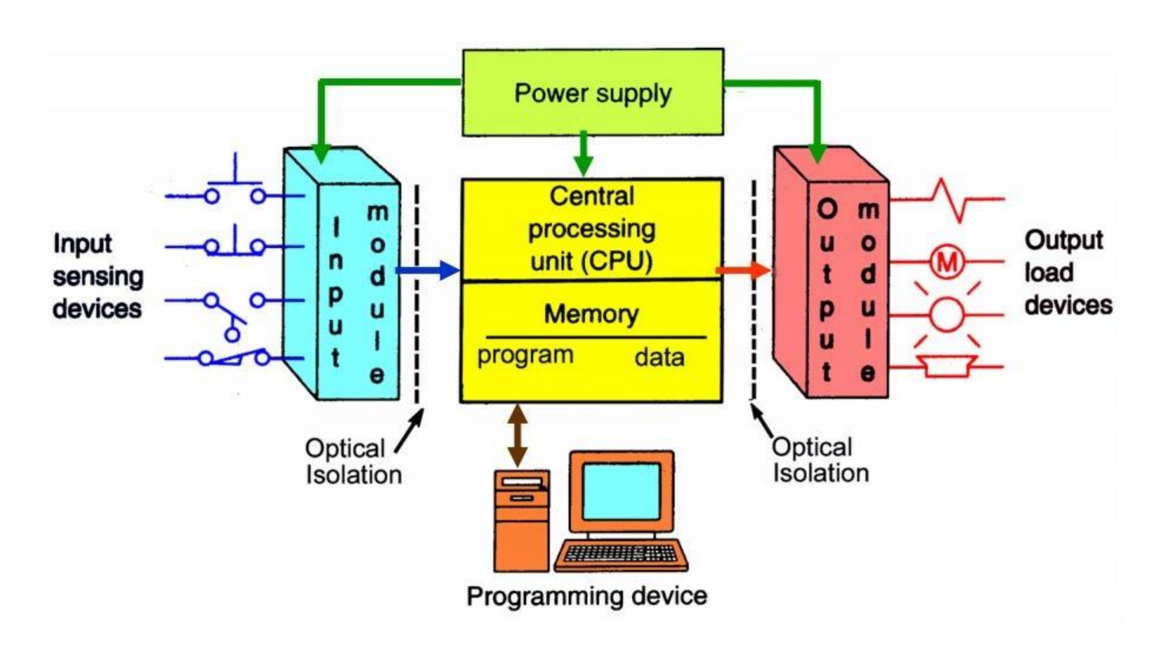
8 - نظام مراقبة و توثيق فوري و مستمر.

- عيوب الـ PLC

1 - بعض التطبيقات يكون استخدام الـ PLC قليل النفع مثل التطبيقات الثابتة التى لا تتغير وبالتالى يكون استخدام الـ PLC بتكلفة عالية او بامكانيات غير مستغلة

2 - ادخال التقنية الحديثة يصاحبها بعض المشاكل كإحداث بعض التقنيات فى الالات واعادة تدريب بعض المهندسين لمواكبة هذا التغيير والتدريب على لغات البرمجة الحديثة

- مكونات الPLC




يتألف جهاز التحكم المنطقي القابل للبرمجة من الوحدات الاساسية التالية:

- وحدات الادخال يتم توصيل وحدة الدخل بمجموعة من العناصر الفيزيائية مثل المفاتيح الكهربائية و المجسات و مقاييس الحرارة و الوزن و مجسات مستوى السوائل و غيرها حيث تقوم وحدة الدخل بإستقبال الأشارات التماثلية و الرقمية المرسلة من هذه العناصر و تقوم بتحويلها إلى إشارات منطقية يمكن ان تتعامل معها وحدة المعالجة المركزية.

وحدات إدخال رقمي :Digital Input Cards تؤدي نفس الوظيفة التي تقوم بها الريليهات ومفاتيح التلامس بحيث أنها تشبه حالات التشغيل والإيقاف ولكن بأسلوب منطقي اخر (إشارة جهد دخل ذو مستوى مرتفع أم منخفض)

وحدات إدخال تماثلي Analog Input Cards: وتشبه إلى حد ما أجهزة الاستشعار أو المجسات حيث يمكن أن تكون لقياس الحرارة، الجهد، أو غير ذلك.

- وحدات إخراج رقمي Digital Output Cards: تؤدي هذه الوحدات عمليات الفتح والقفل مثل المفاتيح العادية ونقاط التلامس المفتوحة والمغلقة في عنصر الريلاي.

وحدات إخراج تماثلي Analog Output Cards: عادة ما يكون خرج هذه الوحدات هو جهد ذا تيار مستمر VDC متغير القيمة ويمكن تشبيهه بمقسم الجهد.

- وحدة المعالجة المركزية(CPU): وهي تماما كتلك الموجودة في الحاسوب الشخصي مع اختلاف بسيط هو قدرتها على تحمل درجات حرارة أعلى نسبيا وذلك حسب المتطلبات الصناعية. تهتم هذه الوحدة بعملية إدارة وتشغيل نظام التشغيل، البرنامج المخزون ومعالجة البيانات بين الدخل، الذاكرة والخرج.

وهي كذلك مركز اتخاذ القرارات لوحدة الـ PLC وتقوم بمايلي:

أستقبال و معالجة الإشارات المنطقية المرسلة من وحدة الدخل

إتخاذ القرارات المناسبة حسب التعليمات المخزنة في ذاكرة البرنامج.

إصدار اوامر التحكم لوحدة الخرج حسب تعليمات البرنامج المخزنة في الذاكرة

تقوم وحدة الـ CPU بعديد من العمليات مثل العد، التوقيت، مقارنة البيانات ، العمليات المتسلسلة و الإزاحة.

- وحدات الذاكرة والتخزين :Memory and Storage هي أيضا شبيها بمحتويات الحاسوب الشخصي مثل ذاكرة الوصول العشوائي ذاكرة الوصول العشوائي ذاكرة القراءة فقط (أصبحت تستبدل ذاكرة القراءة فقط القابلة للبرمجة والمسح ذاكرة القراءة فقط القابلة للبرمجة والمسح أو ذاكرة فلاش .(Flash memory لاتستخدم محركات الأقراص بكافة أنواعها بشكل كبير في هذه الأجهزة وذلك بسبب صغر حجم البرنامج المكتوب.

يوجد نوعين رئيسيين من الذاكرة في وحدة الـ: PLC

الذاكرة العشوائية ( RAM ) وهي الذاكرة التي يمكن إدخال البيانات (DATA ) لها مباشرة من أي عنوان .(Address) كما أنه يمكن كتابة وقراءة البيانات من هذه الذاكرة. وهي ذاكرة غير دائمة أي مؤقتة يعني هذا أن البيانات المخزنة فيها ستفقد في حالة فقد الطاقة الكهربية المشغلة لها و لذلك يتم تركيب بطارية لتجنب فقد البيانات في حالة فقد الطاقة الرئيسية المشغلة لها

ذاكرة القراءة فقط (ROM) وهي الذاكرة التي يمكن قراءة البيانات منها و لكن لا يمكن كتابة البيانات فيها. هذه الذاكرة تستخدم لحماية البيانات أو البرامج المخزنة فيها من المحو، و هي ذاكرة دائمة و هذا يعني أن البيانات المخزنة فيها لن تفقد في حالة فقد الطاقة الكهربائية. تنقسم هذه الذاكرة إلى:

ذاكرة القراءة فقط القابلة للبرمجة و المسح (EPROM) و هي ذاكرة للقراءة فقط و لكن يمكن مسح البيانات منها وذلك بتعريضها للاشعة فوق البنفسجية لتصبح جاهزة لأستقبال بيانات جديدة بواسطة كاتب بيانات خاص بها.

ذاكرة القراءة فقط القابلة للمسح و البرمجة إلكترونياً ( EEPROM) وهي كذلك ذاكرة للقراءة فقط و لكن يمكن ان يتم مسح البيانات المخزنة بها وذلك بوضعها على (صيغة عدم الحماية( Unprotected Mode) و من ثم إدخال بيانات جديدة لها.

- وحدة الاتصال :Communication Card الغرض الرئيسي من هذه الوحدة هو تبادل البيانات بين الجهاز ووحدة أخرى مثل الحاسوب الشخصي أو جهاز PLC اخر.

- وحدة إمداد القدرة وبطارية الدعم Power Supply and Battery Backup: تهتم وحدة إمداد القدرة باستلام القدرة الكهربائية من مصدر خارجي وتوزيعها بشكل ملائم على جميع الوحدات السابقة بجهد مناسب لعملها وتيار مناسب خاصة في وحدات الإخراج التماثلي. أما بطارية الدعم فتستعمل لحفظ نظام التشغيل والبرنامج عند انقطاع الطاقة الكهربائية لفترة من الوقت.

- وحدة المشغلOperator unit

تتيح هذه الوحدة عرض معلومات العمليات المختلفة المتحكم فيها.

- ملحق A0 انواع IC Package Types

| | | |
|---|---|---|
| BGA | Ball Grid Array | |
| BQFP | Bumpered Quad Flat Pack | |
| CBGA | Ceramic Ball Grid Array | |
| CFP | Ceramic Flat Pack | |
| CQFP | Ceramic Quad Flat Pack | |
| TBD | Ceramic Lead-Less Chip Carrier | |
| DLCC | Dual Lead-Less Chip Carrier Ceramic | |
| FBGA | Fine-pitch Ball Grid Array | |
| fpBGA | Fine Pitch Ball Grid Array | |
| JLCC | J-Leaded Chip Carrier Ceramic | |
| LCC | Leaded Chip Carrier also Leadless | |
| LCCC | Leaded Ceramic Chip Carrier | |

| | | |
|---|---|---|
| | Low-Profile, Fine-Pitch Ball Grid Array | **LFBGA** |
| | Land Grid Array Pins are on the Motherboard, not the socket | **LGA** |
| | Micro Leadframe Chip Carrier | **MLCC** |
| | Plastic Ball Grid Array | **PBGA** |
| | Plastic Leaded Chip Carrier | **PLCC** |
| | Plastic Quad Flat Pack | **PQFD** |
| | Plastic Quad Flat Pack | **PQFP** |
| | Plastic Small-Outline Package | **PSOP** |
| | Quad Flat No Leads Package | **QFN** |
| | Quad Flatpack | **QFP** |
| | Quarter Size Outline Package | **QSOP** |
| | Small Outline IC | **SOIC** |
| | Small-Outline Package J-Lead | **SOJ** |
| | Thin Quad Flat Pack | **SSOP** |
| | Thin Small-Outline Package | **TQFP** |

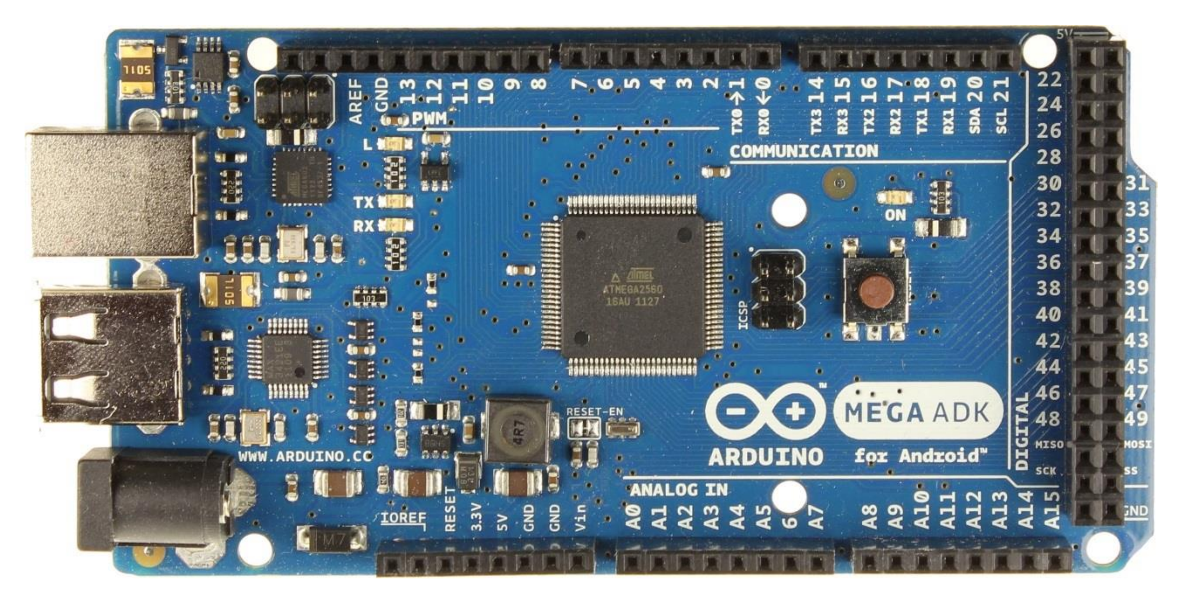
| | | |
|---|---|---|
| | Thin Shrink Small-Outline Package | **TSOP** |
| | Thin Very Small-Outline Package | **TSSOP** |
| | Very-thin Quad Flat Pack | **TVSOP** |
| | Through Hole Device Package Type | **VQFB** |
| | Ceramic Dual In-line Package | **C-DIP** |
| | Ceramic DIP | **CERDIP** |
| | Ceramic Pin Grid Array | **CPGA** |
| | Dual In-line Package | **DIP** |
| | Dual In-line Zig-Zag Package | **TBD** |
| | Hermetic DIP | **HDIP** |
| | Plastic DIP P-DIP | **PDIP** |
| | Pin Grid Array | **PGA** |
| | Plastic Pin Grid Array | **PPGA** |
| | Shrink Dual In-Line Package | **Shrink DIP** |
| | Single In-line Package | **SIP** |
| | single in-line memory module | **simm** |

| | | |
|---|---|---|
| | Programmable Logic Devices | **PLD** |
| | Sidebraze Dual-in-Line Package | **SBDIP** |
| | Small Outline Transistor | **SC-70** |
| | Small Outline Transistor | **SOT-23** |
| | Shrink Plastic Dual-in-Line Package | **SPDIP** |
| | Thin Dual Flat No Leads Package | **TDFN** |
| | Thin Fine-Pitch Ball Grid Array | **TFBGA** |
| | Thin Quad Flat No Leads Package | **TQFN** |
| | Ultra Thin Dual Flat No Leads Package | **UTDFN** |
| | Ultra Thin Quad Flat No Leads Package | **UTQFN** |
| | Very Thin Fine-Pitch Ball Grid Array | **VFBGA** |
| | Very Small Outline Package | **VSOP** |
| | Extreme Thin Dual Flat No Leads Package | **XDFN** |
| | Extreme Thin Quad Flat No Leads Package | **XQFN** |
| | Ceramic Column Grid Array | **CCGA** |

| | | |
|---|---|---|
| | Ceramic Package | **CerPack** |
| | Ceramic Leadless Chip Carrier | **CLCC** |
| | Dou-ble Decawatt Package | **D2PAK or DDPAK** |
| | Decawatt Package 3 | **D3PAK** |
| | Dual Flat No Leads Package | **DFN** |
| | Decawatt Package | **DPAK** |
| | Low-Profile Quad Flat Package | **LQFP** |
| | Micro Leadframe Package | **MLP** |
| | Metric Quad Flat Package | **MQFP** |
| | Micro Small Outline Package | **MSOP** |
| | Power Quad Flat No Leads Package | **PQFN** |
| | Complex programmable logic device | **CPLD** |

- ملحق A انواع لوحات الاردوينو

# أردوينو ميجا اي دي ك(Arduino Mega ADK)




تعتبر الاردوينو اي دي ك من أحد أكبر الأحجام المتوافرة من أنواع الأردوينو, ومع هذه الزيادة في الحجم تأتي الزيادة في الوظائف وعدد المنافذ وحجم الذاكرة وغيرها, فعلى سبيل المثال فإن الاردوينو اونو يحتوي على 14 منفذ رقمي ولكن الاردوينو اي دي ك يحتوي على 54 منفذ رقمي (إدخال/إخراج) اي ما يعادل 4 أضعاف عدد المنافذ المتوافرة في الاردوينو اونو, ولهذا فإننا نلاحظ أن الاردوينو اي دي ك تستخدم نوع اخر من المتحكم الأصغري الذي يستعمل في الاردوينو اونو وهو ATmega2560 ما يميز هذه الاردوينو بأن لديها واجهة مستضيف لوصلة ال USB للسماح للمستخدم بالربط بين لوحة الاردوينو وبين هاتفه النقال الذكي الداعم لنظام اندرويد (android), وذلك بالاعتماد على الدارة المتكاملة (MAX341e) توفرالاردوينو اي دي ك 16 منفذ تماثلي ,(analog input) ويضاف الى ذلك زر إعادة التشغيل, منفذ لوصلة ال USBالخاصة لنقل الطاقة وتحميل نصوص البرمجة من كمبيوترك ورؤوس قابلة للتوصيل تسمح للمستخدم باستخدام برمجة بالتسلسل ,(ICSP HEADER)

الملخص :

| | |
|---|---|
| •المتحكم الأصغري : | ATmega2560 |
| •جهد تشغيل النظام الكهربائي : | v 5 |

•فولطية المنفذ (الموصى به): v 7-12

•فولطية المنفذ (الحد الأقصى والأدني): 6- v 20

•عدد المنافذ الرقمية (إدخال/إخراج) : 54 (15 يمكن استخدامهم كمنافذ للتحكم بالتماثل العرضي للنبضة)

•عدد المنافذ التمثالية (إدخال) : 16

•التيار المستمر لمنفذ 3.3 فولت : mA50

•التيار المستمر لمنفذرقمي (إدخال/إخراج) : mA40

•مساحة الذاكرة : 256 كيلو بايت (8 كيلو بايت تستخدم لمحمّل الإقلاع)

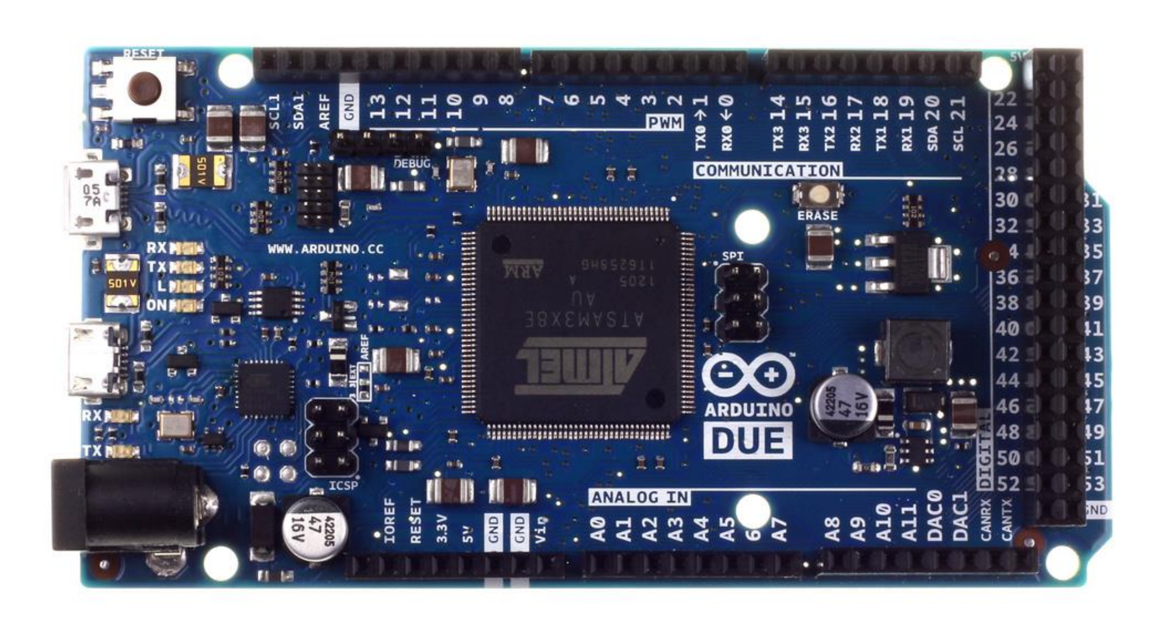
•سرعة المعالج: MHz16

•أبعاد اللوحة : الطول: 4 إنش , العرض: 2.1 إنش

•السعر: $37.40

# الأردوينو ديو(Arduino Due)

الأردوينو ديو هو من أول أنواع الأردوينو التي تستخدم متحكم أصغري ذو 32-بت بنية ARM والذي هو من نوع AT91SAM3X8E مع سرعة معالج تصل الى 84 MHz,قد تختلف الاردوينو ديو عن بقية انواع الاردوينو بأنها تعمل على 3.3 فولت بينما معظم انواع الاردوينو تعمل على 5 فولت, وهذا قد يأثر على قابلية الاردوينو ديو على الارتباط مع بعض انواع الاغطية (shields) وذلك لأن معظم انواع الاغطية المتوافرة تعمل على 5 فولتات بينما الاردوينو ديو يستطيع توفير 3.3 فولتات فقط. الاردوينو ديو لديه 54 منفذ رقمي (إدخال/إخراج) و 12 من هذه المنافذ يمكن استخدامهم كمنافذ إخراج مع تغيير عرض النبضة (PWM), و 12 مدخل إدخال تماثلي, بالإضافة إلى عدد من الرووس والمنافذ ( JATG header, SPI header, TWI, DAC).توفر الاردوينو ديو زر لإعادة التشغيل وزر آخر للمسح والذي لا نراه في كثير من ألواح الأردوينو.




الملخص:

•المتحكم الأصغري: AT91SAM3X8E

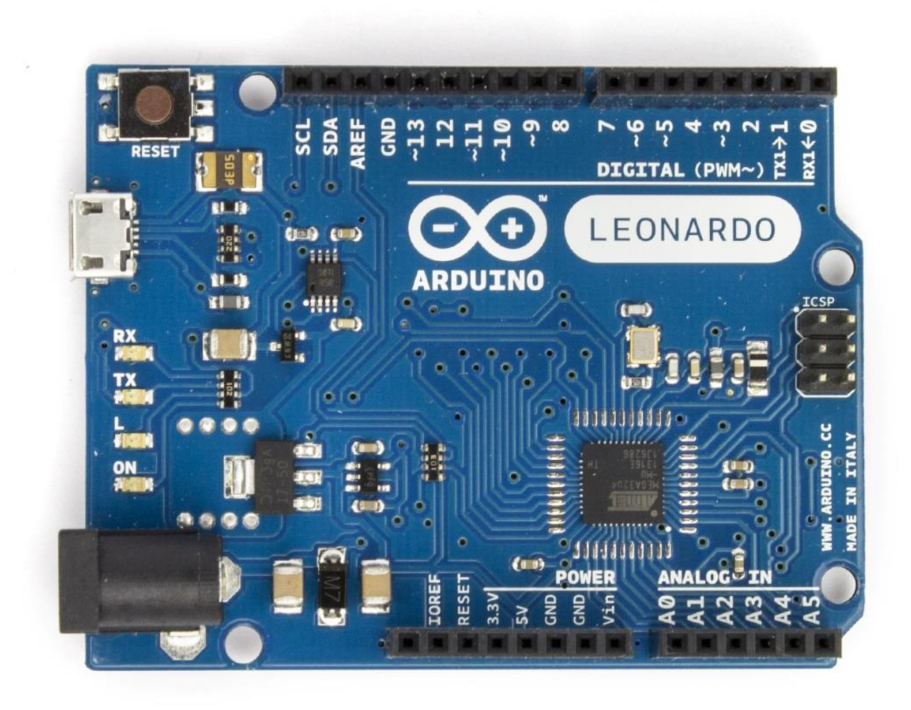
•جهد تشغيل النظام الكهربائي: v 3.3

• فولطية المنفذ (الموصى به): v 7-12

| | |
|---|---|
| •فولطية المنفذ (الحد الاقصى والادني): | 16- 6 v |
| •عدد المنافذ الرقمية (مدخلات/ مخرجات): | 54 (12 يمكن استخدامهم كمنافذ للتحكم بالتماثل العرضي للنبضة) |
| •عدد منافذ الإدخال التماثلية: | 12 |
| •عدد منافذ الإخراج التماثلية: | 2 |
| •التيار المستمر لجميع المنافذ الرقمية (إدخال/إخراج): | mA50 |
| •التيار المستمر لمنفذ 5 فولت: | mA800 |
| •التيار المستمر لمنفذ 3.3 فولت: | mA800 |
| •مساحة الذاكرة: | 512 كيلو بايت متوفرة جميعها للمستخدم |
| •سرعة المعالج: | MHz84 |
| •أبعاد اللوحة : | الطول: 4 إنش, العرض: 2.1 إنش |
| •السعر: | $58.42 |

## أردوينو ليوناردو(Arduino Leonardo)

يعتبر الأردوينو ليوناردو شبيهاً نوعاً ما بالأردوينو اونو , إنما للارديونو ليوناردو ميزات لا يملكها الاردوينو اونو, واحدة من هذه الميزات أن الاردوينو ليوناردو يتسخدم متحكم أصغري مختلف عن الاردوينو اونو, يتسخدم الاردوينو ليوناردو متحكم اصغري من نوعATmega32u4 والذي يحتوي على مخاطب وصلة ال USBمدمج بالمتحكمة, وبذلك فإن لا حاجة لاستخدام معالج اخر (كما نرى في معظم انواع الاردوينو الاخرى) مما يسمح للكمبيوتر بالاتصال بالليوناردو كفأرة حاسوب او كالوحة مفاتيح. ميزة اخرى تحسب لصالح الليوناردو بأن الاردوينو ليوناردو لديه عدد منافذ اكثر, فا لديه 20 منفذ رقمي (إدخال/إخراج) 7 منافذ إخراج منها ذات قدرة على تغيير عرض النبضة, PWM و 12 منفذ تماثلي .(analog input) تحتوي أيضاً لوحة الليوناردو على زر لإعادة التشغيل, منفذ لتوصيل طاقة خارجية, ورؤوس قابلة للتوصيل تسمح للمستخدم بستخدام برمجة بالتتابع ,(ICSP HEADER) ومنفذ لوصلة ال USB المصغرى.(micro USB)

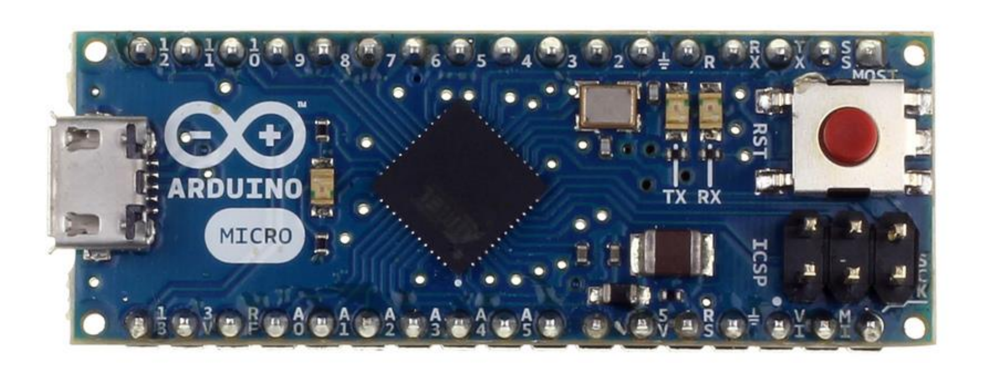


الملخص:

| | |
|---|---|
| •المتحكم الأصغري : | ATmega32u4 |
| •جهد تشغيل النظام الكهربائي : | 5 v |
| • فولطية المنفذ (الموصى به): | 7-12 v |
| •فولطية المنفذ (الحد الأقصى والأدني): | 6- 20 v |
| •عدد المنافذ الرقمية (إدخال/إخراج) : | 20 |
| •منافذ للتحكم بالتماثل العرضي للنبضة: | 7 |
| •عدد المنافذ التماثلية (إدخال) : | 12 |
| •التيار المستمر لمنفذ 3.3 فولت : | 50mA |
| •التيار المستمر لمنفذ رقمي (إدخال/إخراج): | 40mA |
| •مساحة الذاكرة : | 32 كيلو بايت (4 كيلو بايت تستخدم لمحمّل الإقلاع) |
| •سرعة المعالج : | 16MHz |
| •أبعاد اللوحة : | الطول: 2.7 إنش, العرض: 2.1 إنش |
| •السعر: | $26.96 |

# اردوينو ميكرو(Arduino micro)

يعتبر الاردوينو مايكرو الأخ الأصغر للاردوينو ليوناردو فهما يستخدمان نفس المتحكم الأصغريATmega32u4, تحتوي لوحة الارديونو مايكرو على منفذ لوصلة ال USBالمصغرى (micro USB), منفذ لتوصيل طاقة خارجية, ورؤوس قابلة للتوصيل تسمح  للمستخدم باستخدام برمجة بالتسلسل (ICSP HEADER), وزر لإعادة التشغيل. تحتوي أيضاً الاردوينو مايكرو على نفس عدد المنافذ التي توجد الاردوينو ليوناردو ولكن على شكل رؤوس (دبابيس معدنية) وذلك ليتم وضعها في لوحة التجارب (breadboard) بشكلٍ أسهل, تحتوي الاردوينو مايكرو على 20 منفذ رقمي (إدخال/إخراج) و7 منافذ من هذه المنافذ يمكن استخدامهم كمنافذ إخراج مع قدرة تغيير عرض النبضة أو كمنافذ تماثلية (إدخال).



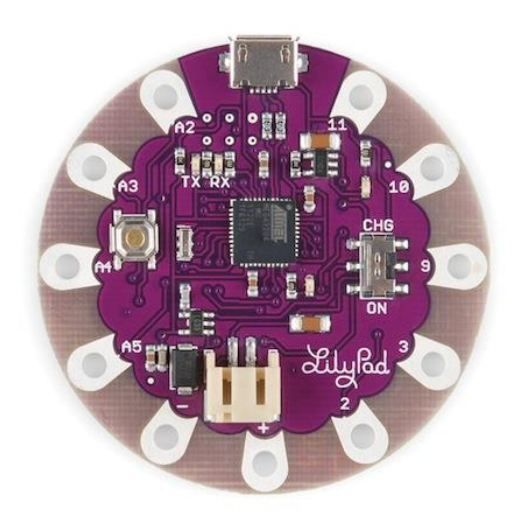

الملخص :

| | |
|---|---|
| •المتحكم الأصغري : | ATmega32u4 |
| •جهد تشغيل النظام الكهربائي : | 5 v |
| • فولطية المنفذ (الموصى به): | 7-12 v |
| •فولطية المنفذ (الحد الأقصى والأدني): | 6- 20 v |
| •عدد المنافذ الرقمية (إدخال/إخراج) : | 20 |

| | |
|---|---|
| •منافذ للتحكم بالتماثل العرضي للنبضة: | 7 |
| •عدد المنافذ التماثلية (إدخال): | 12 |
| •التيار المستمر لمنفذ 3.3 فولت : | mA50 |
| •التيار المستمر لمنفذ (مدخل/مخرج) رقمي: | mA40 |
| •مساحة الذاكرة : | 32 كيلو بايت (4 كيلو بايت تستخدم لمحمّل الإقلاع) |
| •السرعة الميقاتية: | MHz16 |
| •أبعاد اللوحة : | الطول: 4.8 سنتيمتر , العرض: 1.77 سنتيمتر |
| •السعر: | $26.96 |

## ليلي باد اردوينو(LilyPad Arduino USB)

ليلي باد اردوينو تعتبر من أصغر أنواع الاردوينو التي يمكن وصلها بالكمبيوتر عن طريق وصلة ال USB مباشرةً دون الحاجة الى استخدام USB-to-serial adapter) بشكلٍ منفصل, تأخذ الاردوينو ليلي باد الشكل الدائري والذي يصل قطرها الى 5) ٢cm إنش)، توفر الاردوينو ليلي باد ٩ منافذ رقمية (إدخال/إخراج) 4 من هذه يمكن استخدامها كا منفذ إخراج ذو قدرة على تغيير عرض النبضة أو كمنفذ تماثلي (analog input)، لليلي باد أيضاً منفذ لتوصل بطارية ايون الليثيوم ذو 3.7 فولت، ويوجد أيضاً زر لإعادة التشغيل على لوحة الارديونو ليلي باد تستخدم متحكم أضغري من نوع ATmega32u4 على عكس بقية لوح الليلي باد السابقة، تأتي فائدة هذه المتحكمة بأنها تستطيع الاتصال بالكمبيوتر دون الحاجة إلى تعرفيها أي انها سوف تظهرعلى الكمبيوتر كفأرة حاسوب أو كلوحة مفاتيح.



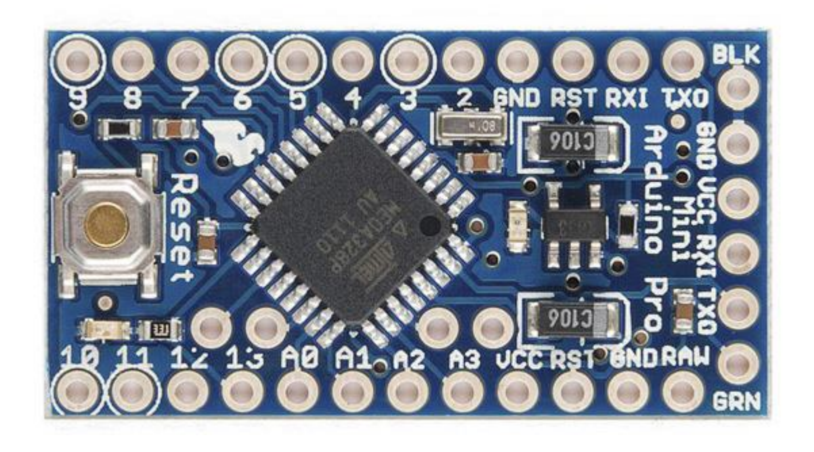

الملخص :

| | |
|---|---|
| •المتحكم الأصغري : | ATmega32u4 |
| •جهد تشغيل النظام الكهربائي : | v 3 |
| •عدد المنافذ الرقمية (إدخال/إخراج) : | 9 |
| •منافذ للتحكم بالتماثل العرضي للنبضة (إخراج): | 4 |
| •عدد المنافذ التماثلية (إدخال) : | 4 |
| •التيار المستمر لمنفذ رقمي (إدخال/إخراج): | mA40 |
| •مساحة الذاكرة : | 32 كيلو بايت (4 كيلو بايت تستخدم لمحمّل الإقلاع) |

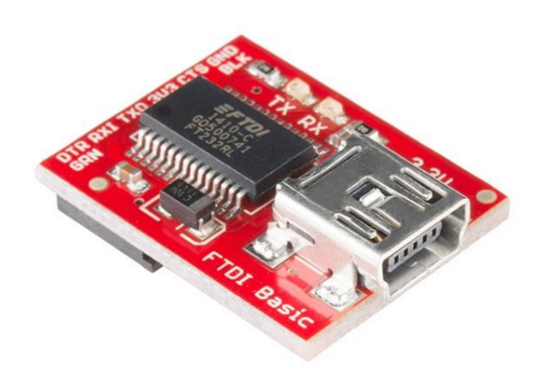
| | |
|---|---|
| •سرعة المعالج: | MHz8 |
| •أبعاد اللوحة : | القطر : 50 ميليمتر |
| •السعر: | $24.95 |

## أردوينو برو ميني(Arduino Pro Mini)

الاردوينو بور ميني هو نوع اخر من انواع الاردوينو صغيرة الحجم, حيث تبلغ أباعد لوحد الميني برو ما يقارب ال0.7×1.3 إنش, تم تصميم الاردوينو برو ميني ليتم وضعه بشكل شبه دائما في المشاريع, ولهذه فإن الاردوينو بور ميني يوجد فيها نوع اخر من انواع المنافذ التي لا نراها كثيراً في بقية أنواع الاردوينو, المنافذ التي توجد في الميني برو صممت ليتم لحمها مع اسلاك خارجية لتوصيل اجزاء خارجية (كا الحساسات او المصابيح او اي اجهزة اخرى). يوجد نوعان من أنواع الاردوينو برو ميني, نوع يعمل على 3.3 فولت و بسرعة 8 MHzواخر يعمل على 5 فولت و بسرعة 16 .MHzكما نلاحظ بالصورة انه لا يوجد على لوحة الاردوينو برو ميني منفذ لوصلة ال USBوالتي تعتبر مهمة من اجل برمجة الاردوينو عن طريق كمبيوترك, ولكن باستخدام الستة الرؤوس المخصصة لوصلة ال FTDI أو باستخدام اللوحة المخصصة من sparkfun يمكنك ايصال الاردوينو بور ميني بالكمبيوتر عن طريق هذه اللوحة أو الوصلة التي ذكرنها إما بهدف البرمجة أو للتزويد الطاقة. توفر الاردوينو برو ميني أيضاً عدداً من المنافذ الرقمية والتماثلية وأيضاً زر لإعادة التشغيل.



## اللوحة الخاصة

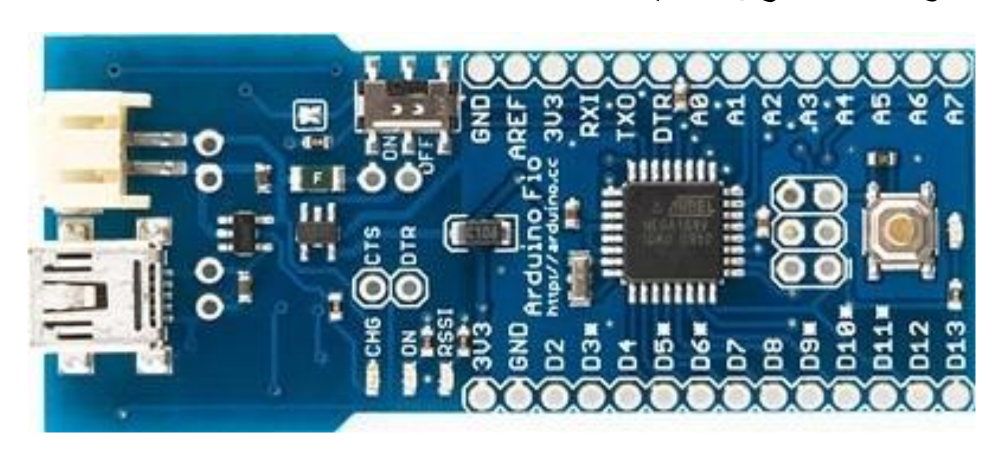



الملخص :

| | |
|---|---|
| •المتحكم الأصغري : | ATmega168 |
| •جهد تشغيل النظام الكهربائي: | 3.3 v او 5v (بناءً على النوع) |
| •عدد المنافذ الرقمية (إدخال/إخراج) : | 14 (6 يمكن استخدامهم كمنافذ للتحكم بالتماثل العرضي للنبضة) |
| •عدد المنافذ التماثلية (إدخال) : | 8 |
| •التيار المستمر لمنفذ (مدخل/مخرج) رقمي: | 40mA |
| •مساحة الذاكرة : | 16 كيلو بايت (2 كيلو بايت تستخدم لمحمّل الإقلاع) |
| •سرعة المعالج (لنوع ال 3.3 فولت ): | 8MHz |
| •سرعة المعالج (لنوع ال 5 فولت ): | 16MHz |
| •أبعاد اللوحة : | الطول: 0.7 إنش , العرض: 1.3 إنش |
| •السعر: | $9.95 |

# أردوينو فيو(Arduino Fio)

إن الاردوينو فيو هي أحد أنواع الاردوينو المعدة والمخصصة للتطبيقات الاسلكية, فنلاحظ أن الاردوينو فيو تحتوي على منافذ خاصة تسمح للمستخدم بإيصال رقاقة ال XBeeمما يمكن الاردوينو من استقبال وإرسال المعلومات مع رقاقة XBee أخرى لاسلكياً, هذه الخاصية تسمح للمستخدم بتحميل نصوص البرمجة لاسلكياً على الاردوينو فيو باستخدام .(USB-to-XBee adaptor) اما إذا أردت تحميل نصوص البرمجة بالطريقة المعتادة (اي توصليها بكومبيوترك عن طريق وصلة ال USB(فهناك طريقتان, إما أن تستخدم وصلة ال FTDI أو أن تستخدام اللوحة المخصصة من sparkfunأما بالنسبة لمكونات لوحة الاردوينو فيو فاللوحة تحتوي على منفذ لتوصيل بطارية ليثيوم خارجية منفصلة, زر لإعادة التشغيل, و 14 منفذ رقمي (إدخال/إخراج) 6 من هذه المنافذ يمكن استخدامهم كمنافذ إخراج مع قدرة تغيير عرض النبضة PWM,و8 منافذ اخرى تماثلية.

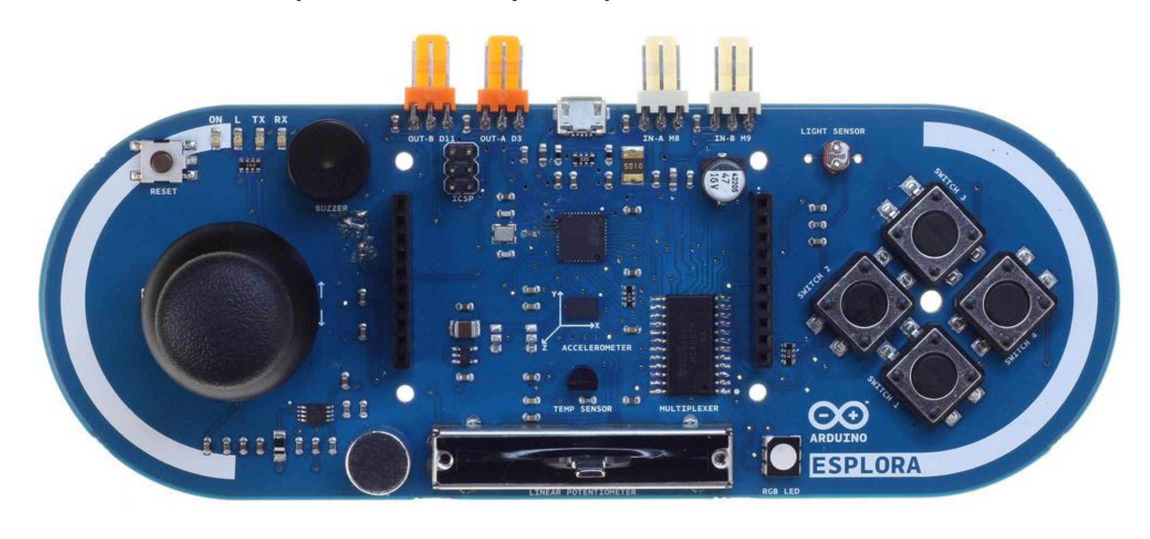


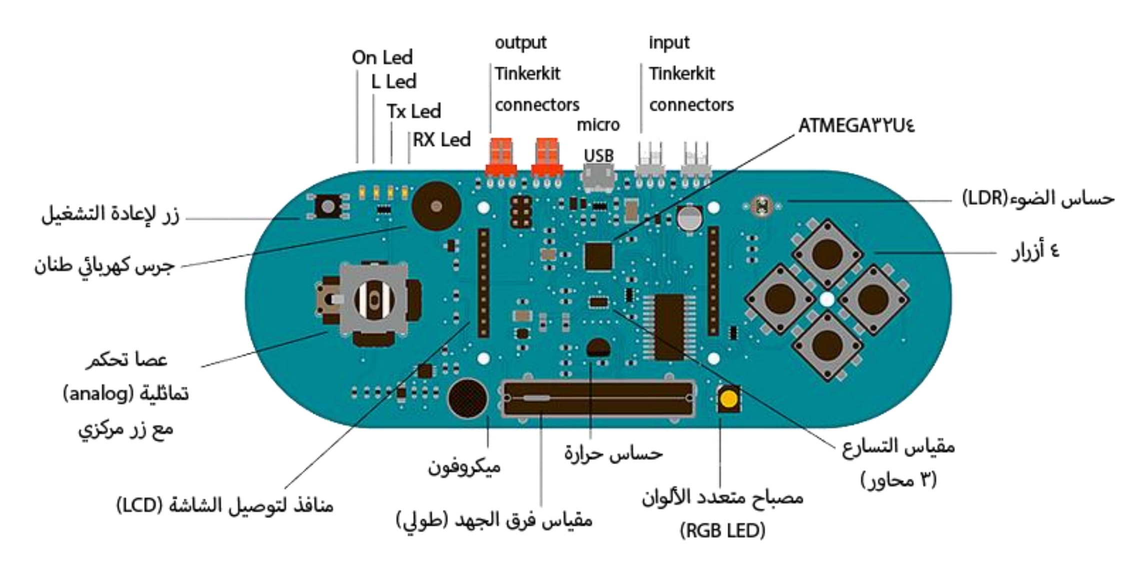

الملخص :

| | |
|---|---|
| •المتحكم الأصغري: | ATmega328p |
| •جهد تشغيل النظام الكهربائي: | v 3.3 |
| • فولطية المنفذ (الموصى به): | v 3.35-12 |
| •فولطية المنفذ (الحد الأقصى والأدني): | v 3.7-7 |

| | |
|---|---|
| •عدد المنافذ الرقمية (إدخال/إخراج): | 14 (6 يمكن استخدامهم كمنافذ للتحكم بالتماثل العرضي للنبضة) |
| •عدد امنافذ الإدخال التماثلي: | 8 |
| •التيار المستمر لمنفذ رمقي(إدخال/إخراج): | mA50 |
| •مساحة الذاكرة: | 32 كيلو بايت (4 كيلو بايت تستخدم لمحمّل الإقلاع) |
| •سرعة المعالج: | MHz8 |
| •أبعاد اللوحة: | الطول: 1.1 إنش, العرض: 2.6 إنش |
| •السعر: | $24.95 |

# أردوينو إسبلورا (Arduino Esplora)




تعتبر الاردوينو اسبلورا مستمدة من الاردوينو ليوناردو, فهما يستخدمان نفس نوع المتحكم الأصغريATmega32u4 , ومع ذلك فإننا نرى اختلافا كبيرا بينهما, فقد تم تصميم الارديونو اسبلورا بهدف السماح للمستخدم بالمباشرة بتصميم المشاريع واستخدام الاردوينو مباشرةً دون الحاجة الى تعلم عن مبادئ الإلكترونيات وكيفية توصيلها وكثير من الامور الاخرى التي قد تقف عائق في طريق المستخدم, ولهذا فإن لوحة الاردوينو اسبلورا تحتوي على الكثير من الازرة والمصابيح والحساسات وغيرها من الاجهزة الجاهزة للاستخدام والبرمجة (أنظر الى الصورة في الاسفل لتعرف المزيد

عن ما تحتويه الاردوينو اسبلورا من اجهزة), تبلغ سرعة المعالج للاردوينو اسبلورا 16 MHzوتوفر ال ATmega32u4 مساحة ذاكرة بحجم 32 كيلو بايت (4 كيلو بايت من هذه المساحة تستخدم لمحمّل الإقلاع), أما بالنسبة الى جهد تشغيل النظام الكهربائي فهو 5 فولت. يبلغ طول الاردوينو اسبلورا 6.5 إنش ويصل عرضها الى 2.4 إنش. أما بالنسبة إلى سعر الاردوينو اسبلورا فهو 62.76$.

## أردوينو روبوت(Arduino Robot)

الاردوينو روبوت هو اول اردوينو قادر على الحركة بواسطة العجلات وليس هذا فحسب بل انه قادر على تحديد مساره بنفسه واكثر من ذلك بكثير اذا ما قمت ببرمجته بشكل صحيح وفعلا, كما قلنا من قبل فإن إمكانيات لوحة الاردوينو لا حدود لها, فكيف اذا كان هذه الاردوينو عبارة عن جهاز يحتوي على لوحتان من الاردوينو ليوناردو وتحتوي على الحساسات والأجهزة المختلفة وقادرة على الحركة.!

كما نعلم فإن الاردوينو مفتوحة المصدر, مما يعني انه يمكنك تعلم بالتحديد كيف الاردوينو روبوت يعمل واستخدامه ليكون نقطة البداية لروبوتك الخاص .

للاردوينو روبوت معالجان, معالج في اللوحة السفلى ومعالج في اللوحة العليا,اللوحة السفلى مسؤولة عن التحكم بالمحركات ولهذه فإنها تسمى لوحة المحركات (Motor Board),واللوحة العليا تسمى لوحة السيطرة (Control Borad) وهي مسؤولة عن قراءة الحساسات وتحديد كيفية العمل. كلا اللوحتان يستخدمان المتحكم الأصغري من نوع ATmega32u4 وكلاهما ايضا يمكن برمجتهم باستخدام برنامج الاردوينو الخاص بالكمبيوترات.(Arduino IDE) اما بالنسبة لسعر الاردوينو روبوت فتبلغ قميته $283.11.



الطاقة

يمكن تزويد اردوينو روبوت بالطاقة اما عن طريق وصلة ال USB او عن طريق 4 بطاريات من حجم AA, يتم تحديد مصدر الطاقة أوتماتيكيا.

حاملة البطاريات تتسع ل 4 بطاريات من حجم AA القابلة لإعادة الشحن (لا تستخدم بطاريات غير قابلة للشحن).

من أجل السلامة, يتم تعطيل المحركات عندما يكون مصدر الطاقة أتي من وصلة ال USB.

يحتوي الارديونو روبوت على شاحن لهذه البطاريات والذي يتطلب مصدر طاقة بتيار ثابت بقيمة 9 فولتات قادم من محول يتم وضعه في الحائط (AC-to-DC wall adapter)يتم توصيله بمنفذ خاص موجود على لوحة المحركات, الشاحن لن يعمل اذا كانت وصلة ال USB موصولة بالارديونو روبوت.

كلاً من لوحة التحكم ولوحة المحركات تستمد الطاقة من مصدر الطاقة الموجود على لوحة المحركات.

الذاكرة

للمتحكم الأصغري ATmega32u4 مساحة ذاكرة تصل الى 32 كليو بايت و 4 كيلو بايت من هذه المساحة يتم استخدامها من اجل محمُل الإقلاع.

يوجد على لوحة التحكم قارئ لبطاقة ذاكرة خارجية (SD card reader) مما يسمح بإضافة مساحة إضافية للتخزين.

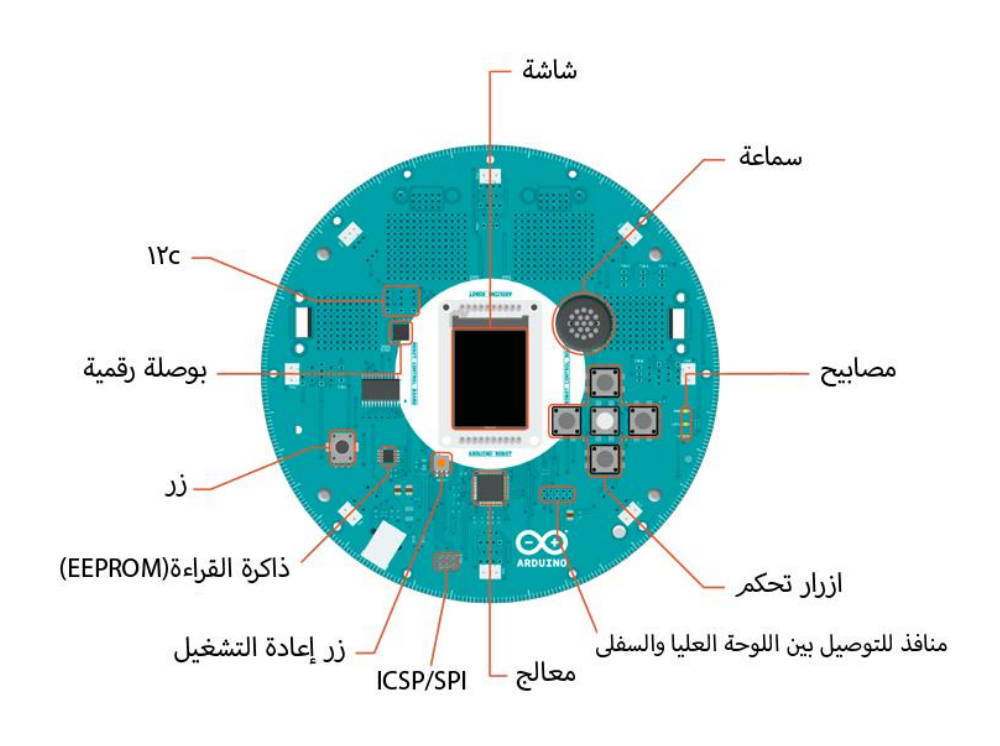
الخواص الفيزيائية

يبلغ قطر الارديونو روبوت 19 سنتيمتر, ويصل طوله الى 10 سنتيمتر مع المكونات التي توجد على كلا اللوحتان.

ملخص لوحة التحكم :

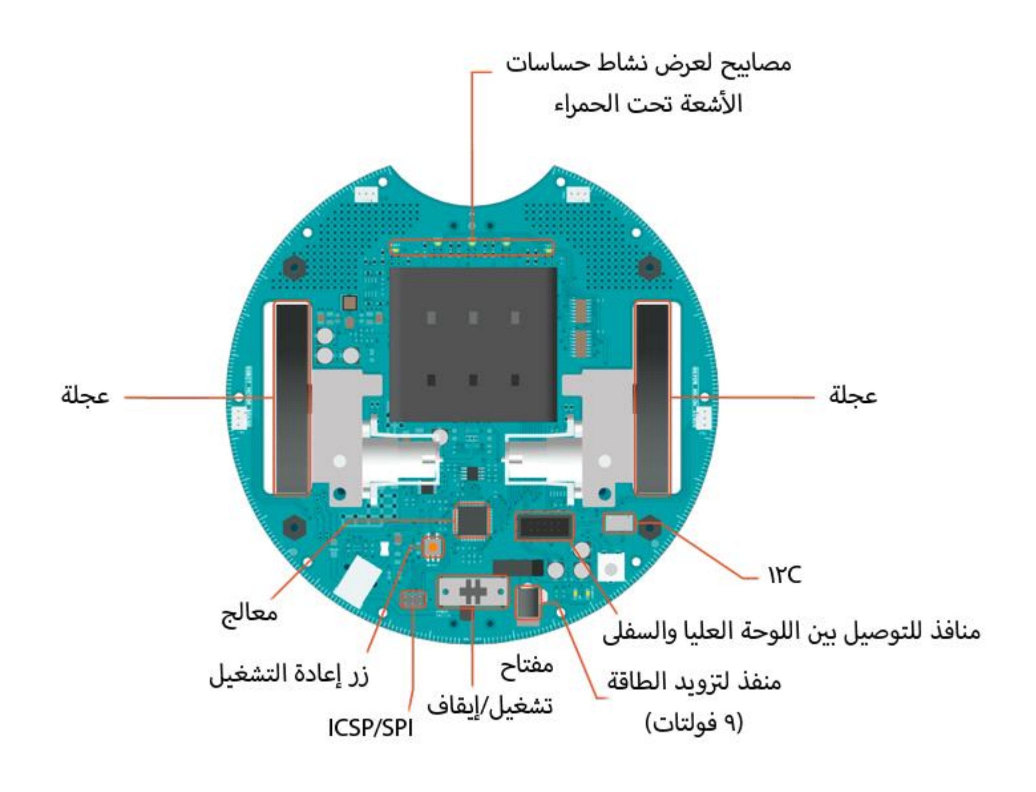
| | |
|---|---|
| المتحكم الأصغري : | ATmega32u4 |
| جهد تشغيل النظام الكهربائي : | v5 |
| عدد المنافذ الرقمية (إدخال/إخراج) : | 5 |

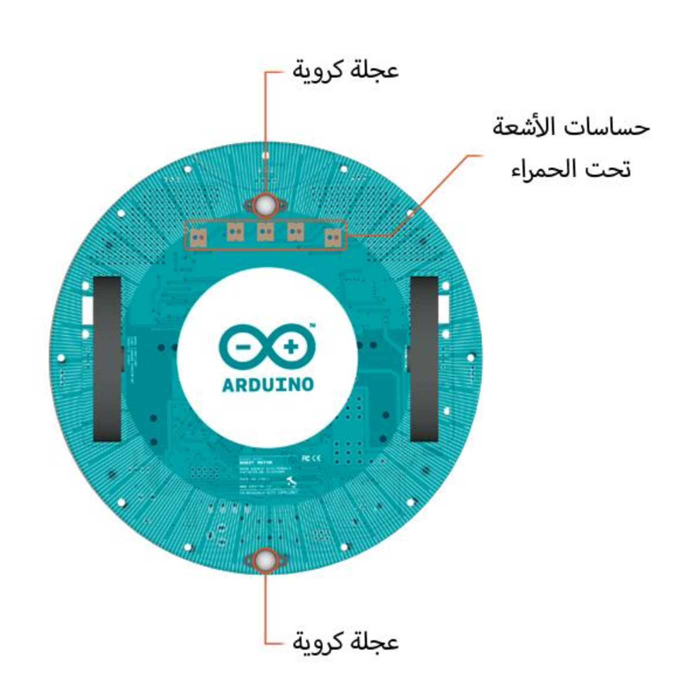
| | |
|---|---|
| عدد منافذ الإخراج ذو تيار متردد: | 6 |
| عدد المنافذ التماثلية: | 4 (من المنافذ الرقمية (إدخال/إخراج)) |
| عدد المنافذ التماثلية (متعددة الاتصال): | 8 |
| التيار المستمر لمنفذ رقمي (إدخال/إخراج) : | mA40 |
| مساحة الذاكرة : | 32 كيلو بايت (4 كيلو بايت تستخدم لمحمّل الإقلاع) |
| سرعة المعالج: | MHz16 |
| عدد الازرة : | 5 |
| السماعة : | Ohm8 |
| قارئ بطاقات الذاكرة: | يتم استخدامه لبطاقة ذاكرة من نوع FAT16 مهيئة |
| c12 منافذ اللحام: | 3 |
| مناطق لعمل نوذج أولي : | 4 |



## ملخص لوحة المحركات :

| | |
|---|---|
| المتحكم الأصغري : | ATmega32u4 |
| جهد تشغيل النظام الكهربائي : | v5 |
| عدد المنافذ الرقمية (إدخال/إخراج) : | 4 |
| عدد منافذ الإخراج ذو تيار متردد: | 1 |
| عدد المنافذ التماثلية: | 4 من المنافذ الرقمية (إدخال/إخراج) |
| التيار المستمر لمنفذ رقمي(إدخال/إخراج) : | mA40 |
| مساحة الذاكرة : | 32 كيلو بايت (4 كيلو بايت تستخدم لمحمّل الإقلاع) |
| سرعة المعالج : | MHz16 |
| c12 منافذ اللحام: | 1 |
| مناطق لعمل نوذج أولي : | 2 |




عجلة كروية
حساسات الأشعة
تحت الحمراء
ARDUINO
عجلة كروية

يوضح هذا الجدول مقارنة سريعة بين خصائص جميع لوحات Arduino

| Name | Processor | Operating/Input Voltage | CPU Speed | Analog In/Out | Digital IO/PWM | USB |
|---|---|---|---|---|---|---|
| 101 | Intel® Curie | 3.3 V/ 7-12V | 32MHz | 6/0 | 14/4 | Regular |
| Gemma | ATtiny85 | 3.3 V / 4-16 V | 8 MHz | 1/0 | 3/2 | Micro |
| LilyPad | ATmega168V ATmega328P | 2.7-5.5 V / 2.7-5.5 V | 8MHz | 6/0 | 14/6 | - |
| LilyPad SimpleSnap | ATmega328P | 2.7-5.5 V / 2.7-5.5 V | 8 MHz | 4/0 | 9/4 | - |
| LilyPad USB | ATmega32U4 | 3.3 V / 3.8-5 V | 8 MHz | 4/0 | 9/4 | Micro |
| Mega 2560 | ATmega2560 | 5 V / 7-12 V | 16 MHz | 16/0 | 54/15 | Regular |
| Micro | ATmega32U4 | 5 V / 7-12 V | 16 MHz | 12/0 | 20/7 | Micro |
| MKR1000 | SAMD21 Cortex-M0+ | 3.3 V/ 5V | 48MHz | 7/1 | 8/4 | Micro |
| Pro | ATmega168 ATmega328P | 3.3 V / 3.35-12 V 5 V / 5-12 V | 8 MHz 16 MHz | 6/0 | 14/6 | - |
| Pro Mini | ATmega328P | 3.3 V / 3.35-12 V 5 V / 5-12 V | 8 MHz 16 MHz | 6/0 | 14/6 | - |
| Uno | ATmega328P | 5 V / 7-12 V | 16 MHz | 6/0 | 14/6 | Regular |
| Zero | ATSAMD21G18 | 3.3 V / 7-12 V | 48 MHz | 6/1 | 14/10 | 2 Micro |
| Due | ATSAM3X8E | 3.3 V / 7-12 V | 84 MHz | 12/2 | 54/12 | 2 Micro |
| Esplora | ATmega32U4 | 5 V / 7-12 V | 16 MHz | - | - | Micro |
| Ethernet | ATmega328P | 5 V / 7-12 V | 16 MHz | 6/0 | 14/4 | Regular |
| Leonardo | ATmega32U4 | 5 V / 7-12 V | 16 MHz | 12/0 | 20/7 | Micro |
| Mega ADK | ATmega2560 | 5 V / 7-12 V | 16 MHz | 16/0 | 54/15 | Regular |
| Mini | ATmega328P | 5 V / 7-9 V | 16 MHz | 8/0 | 14/6 | - |
| Nano | ATmega168 ATmega328P | 5 V / 7-9 V | 16 MHz | 8/0 | 14/6 | Mini |
| Yùn | ATmega32U4 AR9331 Linux | 5 V | 16 MHz 400MHz | 12/0 | 20/7 | Micro |
| Arduino Robot | ATmega32u4 | 5 V | 16 MHz | 6/0 | 20/6 | 1 |
| MKRZero | SAMD21 Cortex-M0+ 32bit low | 3.3 V | 48 MHz | 7 (ADC 8/10/12 bit)/1 | 22/12 | 1 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | power ARM MCU | | | (DAC 10 bit) | | |

# Arduino Retired boards specs

| **Name** | Processor | Operating/Input Voltage | CPU Speed | Analog In/Out | Digital IO/PWM | USB |
|---|---|---|---|---|---|---|
| **BT** | ATmega328P | 5 V / 2.5-12 V | 16 MHz | 6/0 | 14/6 | - |
| **Fio** | ATmega328P | 3.3 V / 3.7-7 V | 8 MHz | 8/0 | 14/6 | Mini |

## الكلمات المحجوزة في لغة C Arduino

### Digital I/O

digitalRead()
digitalWrite()
pinMode()

### Analog I/O

analogRead()
analogReference()
analogWrite()

### Zero, Due & MKR Family

analogReadResolution()
analogWriteResolution()

### Advanced I/O

noTone()
pulseIn()
pulseInLong()
shiftIn()
shiftOut()
tone()

### Time

delay()
delayMicroseconds()
micros()
millis()

### Math

abs()
constrain()
map()
max()
min()
pow()
sq()
sqrt()

### Trigonometry

cos()
sin()
tan()

### Characters

isAlpha()
isAlphaNumeric()
isAscii()
isControl()
isDigit()
isGraph()
isHexadecimalDigit()
isLowerCase()
isPrintable()
isPunct()
isSpace()
isUpperCase()
isWhitespace()

Random Numbers

random()
randomSeed()

Bits and Bytes

bit()
bitClear()
bitRead()
bitSet()
bitWrite()
highByte()
lowByte()

External Interrupts

attachInterrupt()
detachInterrupt()

Interrupts

interrupts()
noInterrupts()

Communication

Serial
Stream

USB

Keyboard
Mouse

VARIABLES

Arduino data types and constants.

Constants

Floating Point Constants
Integer Constants
HIGH | LOW
INPUT | OUTPUT | INPUT _PULLUP
LED_BUILTIN
true | false

Conversion

byte()
char()
float()
int()
long()
word()

Data Types

String()
array
bool
boolean
byte
char
double
float
int
long
short
string
unsigned char
unsigned int
unsigned long
void
word

Variable Scope & Qualifiers

const
scope
static
volatile

Utilities

PROGMEM
sizeof()

# STRUCTURE

The elements of Arduino (C++) code.

## Sketch

loop()
setup()

## Control Structure

break
continue
do...while
else
for
goto
if...else
return
switch...case
while

## Further Syntax

#define (define)
#include (include)
/* */ (block comment)
// (single line comment)
; (semicolon)
{} (curly braces)

## Arithmetic Operators

% (remainder)
* (multiplication)
+ (addition)
- (subtraction)
/ (division)
= (assignment operator)

## Comparison Operators

!= (not equal to)
< (less than)
<= (less than or equal to)
== (equal to)
> (greater than)
>= (greater than or equal to)

## Boolean Operators

! (logical not)
&& (logical and)
|| (logical or)

## Pointer Access Operators

& (reference operator)
* (dereference operator)

## Bitwise Operators

& (bitwise and)
<< (bitshift left)
>> (bitshift right)
^ (bitwise xor)
| (bitwise or)
~ (bitwise not)

Compound Operators

&= (compound bitwise and)
*= (compound multiplication)
++ (increment)
+= (compound addition)
-- (decrement)
-= (compound subtraction)
/= (compound division)
^= (compound bitwise xor)
|= (compound bitwise or)

- ملحق A3 – قطع يمكن استخدامها مع الاردوينو

# VOICE RECOGNITION MODULE



Description:

Voice Recognition Module V3
Speak to Control (Arduino compatible)

Overview:
ELECHOUSE Voice Recognition Module is a compact and easy-control speaking recognition board.

This product is a speaker-dependent voice recognition module. It supports up to 80 voice commands in all.

Max 7 voice commands could work at the same time. Any sound could be trained as command. Users need

to train the module first before let it recognizing any voice command.

This board has 2 controlling ways: Serial Port (full function), General Input Pins (part of function). General

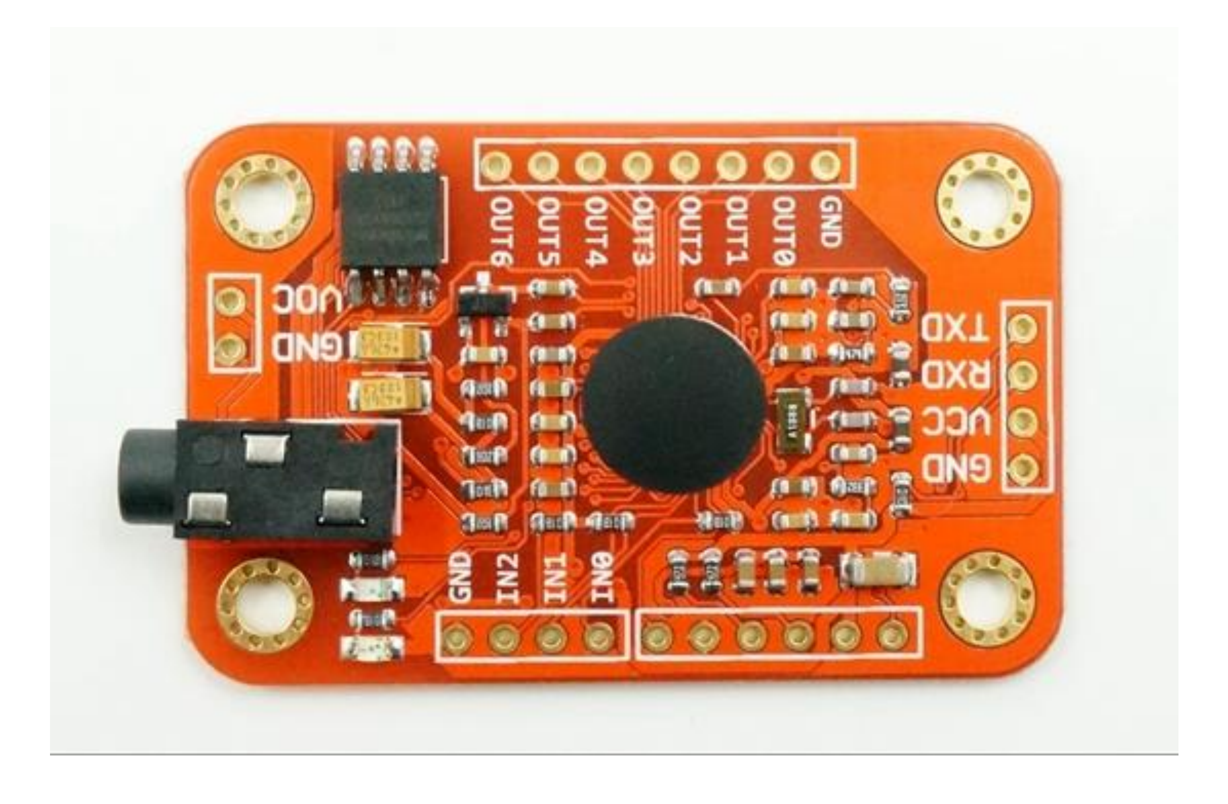
Output Pins on the board could generate several kinds of waves while corresponding voice command was

.recognized

Parameter:
Voltage: 4.5-5.5V
Current: <40mA
Digital Interface: 5V TTL level for UART interface and GPIO
Analog Interface: 3.5mm mono-channel microphone connector + microphone pin interface
Size: 31mm x 50mm
Recognition accuracy: 99% (under ideal environment

Features:

Support maximum 80 voice commands, with each voice 1500ms (one or two words speaking)
Maximum 7 voice commands effective at same time
Arduino library is supplied
Easy Control: UART/GPIO
User-control General Pin Output



# OPTICAL FINGERPRINT READER SENSOR MODULE




Description:

Secure your project with biometrics - this all-in-one optical fingerprint sensor will make adding fingerprint detection and verification super simple. These modules are typically used in safes - there's a high powered DSP chip that does the image rendering, calculation, feature-finding and searching. Connect to any microcontroller or system with TTL serial, and send packets of data to take photos, detect prints, hash and search. You can also enroll new fingers directly - up to 1000 finger prints can be stored in the onboard FLASH memory. There's a green LED in the lens that lights up during a photo so you know its working.

We like this particular sensor because not only is it easy to use, it also comes with fairly straight-forward Windows software that makes testing the module simple - you can even enroll using the software and see an image of the fingerprint on your computer screen. But, of course, we wouldn't leave you a datasheet and a "good luck!" - we wrote a full Arduino library so that you can get running in under 10 minutes. The library can enroll and search so its perfect for any project. We've also written a detailed tutorial on wiring and use. This is by far the best fingerprint sensor you can get.

Usage is very simple : as long as there are serial microcontroller , embedded modules can operate this , MSP430, 51, AVR, PIC, ARM, FPGA,Raspberry Pi, Arduino controller so you can operate this module . This module is controlled through the serial port , you can use the computer's serial port to control this module.
Products from optical fingerprint sensor , high-speed DSP processors , high-performance fingerprint matching algorithms, such as large -capacity FLASH chip hardware and software composition. The fingerprint module is stable, functional, both fingerprint , a variety of functions fingerprint registration , fingerprint matching , fingerprint search.

Application: widely used fingerprint modules , fingerprint identification systems suitable for all from high-end to low-end . For example:
Security fingerprint door locks , safes, gun cases, and finance ;
Access control systems , industrial machines , POS machines, driver training , attendance and other identity areas ;
Private club management , management software , licensing, and so on.

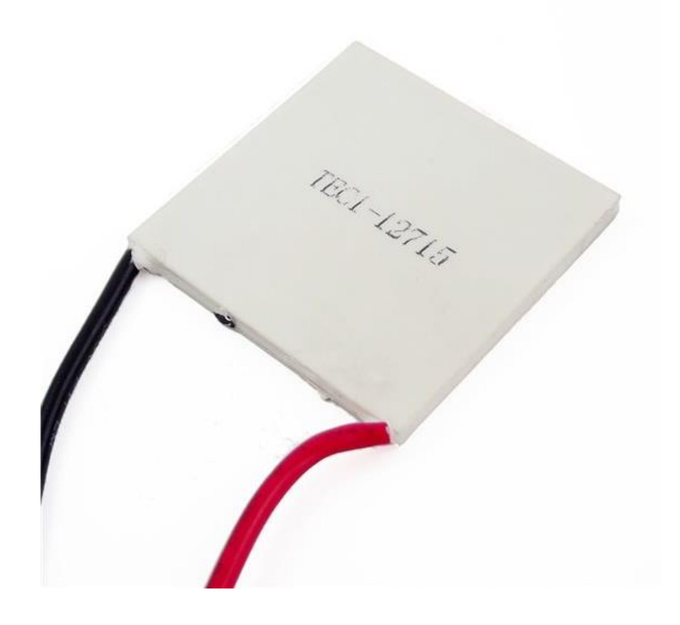
Overview
Secure your project with biometrics - this all-in-one optical fingerprint sensor will make adding fingerprint detection and verification super simple. These modules are typically used in safes - there's a high powered DSP chip that does the image rendering, calculation, feature-finding and searching. Connect to any microcontroller or system with TTL serial, and send packets of data to take photos, detect prints, hash and search. You can also enroll new fingers directly - up to 1000 finger prints can be stored in the onboard FLASH memory. There's a green LED in the lens that lights up during a photo so you know its working.

We like this particular sensor because not only is it easy to use, it also comes with fairly straight-forward Windows software that makes testing the module simple - you can even enroll using the software and see an image of the fingerprint on your computer screen. But, of course, we wouldn't leave you a datasheet and a "good luck!" - we wrote a full Arduino library so that you can get running in under 10 minutes. The library can enroll and search so its perfect for any project. We've also written a detailed tutorial on wiring and use. This is by far the best fingerprint sensor you can get.

Enrolling vs. Searching

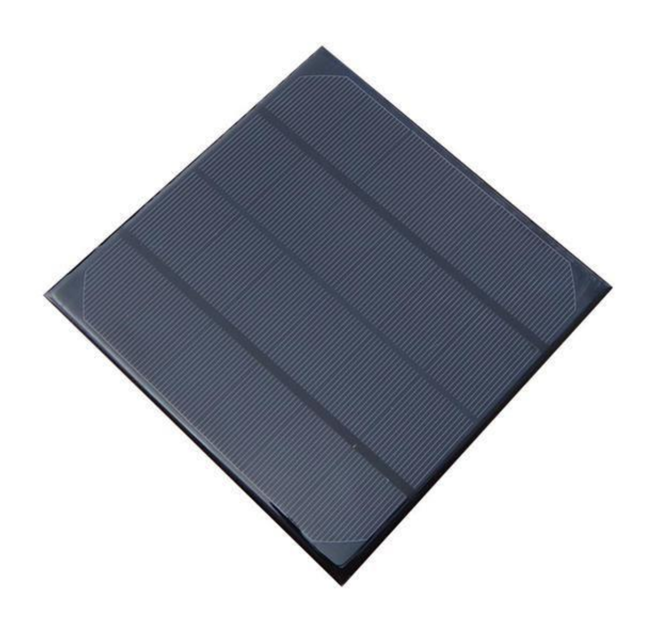
There are basically two requirements for using the optical fingerprint sensor. First is you'll need toenroll fingerprints - that means assigning ID #'s to each print so you can query them later. Once you've enrolled all your prints, you can easily 'search' the sensor, asking it to identify which ID (if any) is currently being photographed.
You can enroll using the windows software (easiest and neat because it shows you the photograph of the print) or with the Arduino sketch (good for when you don't have a windows .(machine handy or for on-the-road enrolling

# PELTIER THERMOELECTRIC COOLER 12V 40MM




Data sheet :
http://www.thermonamic.com/TEC1-12715-English.PDF

# SOLAR PANELS HIGH EFFICIENCY 6V 4.5W 520MAH (165X165)MM



Description:

Power: 4.5W
Voltage: 6V
Material:Polycrystalline
Size: 16.5x16.5x0.3cm
Weight: 100g
Features:

6V 4.5W Solar Panel
High conversion ratehigh efficiency output
Excellent weak light effect
It is important electric parts for transferring the sun light power into electrical power for your application
Build your DIY powered modelssolar displaysolar light and solar toysetc
Usage:To charge cell phones.
For home lighting.
Suitable for solar powered water pumpsmall solar power systemDC Charger etc.
For DIY solar power toys.

# SOLAR CHARGE CONTROLLER REGULATOR BSV20A






Description:

20A 12V 24V intelligent PV solar cell Battery charge controller regulator

Features:

Overcharge voltage protection: 14.4V/28.8V; over discharge protection: 10.8V/21.6V; Max. operation voltage: 25V/36V; Operation temperature: -20~50'C

# JOYSTICK MODULE



Description:

Key FeaturesEasy breadboard connection

Two independent 10K potentiometers with common ground

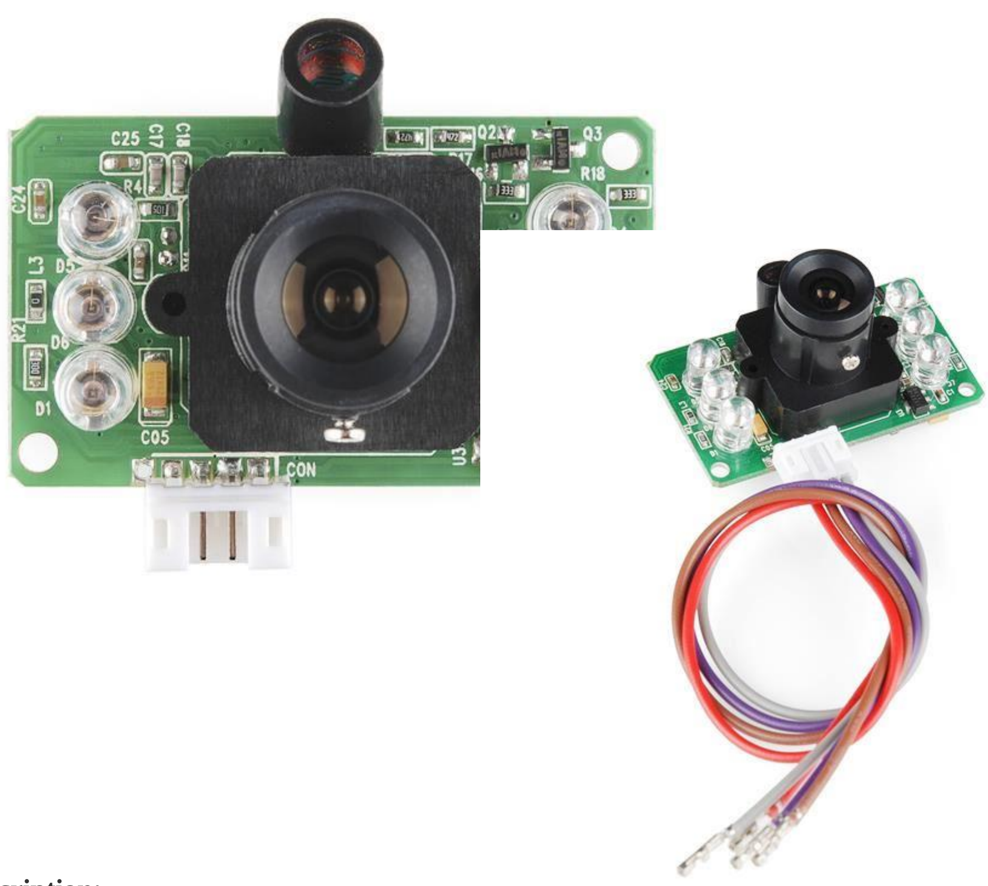
Spring auto-return to center

Comfortable cup-type knob

Specifications Dimensions (cm) Weight (kg)

11 x 8 x 2
0.015

# LINKSPRITE JPEG COLOR CAMERA



**Description:**

LinkSprite JPEG Color Camera TTL Interface - Infrared

Are you a super spy trying to spot ninja assassins in the dark? Or maybe you're just trying to figure out who's been sneaking to the kitchen at night and eating your snacks… Either wayan infrared camera is the way to go! LinkSprite's latest serial port camera module can capture high resolution pictures and transport them over TTL serialmaking it ideal for embedded applications. And it can do it in the darkthanks to infrared LEDs. The infrared feature even has a built-in light sensorso as soon as the ambient light gets low enoughit will automatically turn on the infrared LEDs for night vision!

**Dimensions:** 45.6x30x28mm

**Features:**

- VGA/QVGA/160*120 resolution
- Support capture JPEG from serial port
- Default baud rate of serial port is 38400
- DC 3.3V or 5V power supply

- Current consumption: 80-100mA
- The pin near C03 is AV output pinwhich is an analog output pin

# WATER FLOW SENSOR



Description:

Water flow sensor consists of a plastic valve bodya water rotorand a hall-effect sensor. When water flows through the rotorrotor rolls. Its speed changes with different rate of flow. The hall-effect sensor outputs the corresponding pulse signal. This one is suitable to detect flow in water dispenser or coffee machine.
We have a comprehensive line of water flow sensors in different diameters. Check them out to find the one that meets your need most.

Specifications
Mini. Wokring Voltage: DC 4.5V
Max. Working Current: 15mA (DC 5V)
Working Voltage: DC 5V~24V
Flow Rate Range: 1~30L/min
Load Capacity: ?10mA (DC 5V)
Operating Temperature: ?80?
Liquid Temperature: ?120?
Operating Humidity: 35%~90%RH
Water Pressure: ?1.75MPa
Storage Temperature: -25~+ 80?
Storage Humidity: 25%~95%RH

Features:

CompactEasy to Install
High Sealing Performance
High Quality Hall Effect Sensor
RoHS Compliant

# (DATA LOGGER MODULE(SD CARD




Description:

OpenLog is an open source data logger. Simple to usesimple to change. We wanted to create a serial logger that just worked. Power up this device and it will start logging any received serial data at 9600bps. Sending Ctrl+z three times will drop out of logging and in to command mode. new  will create a new file.  md  makes a directory? brings up the list of commands. OpenLog doesn't do a lotbut it does log serial streams extremely well. Example:

That s really it! OpenLog firmware is open source and is based on Bill Greimans sdfatlib. OpenLog currently supports FAT16 and FAT32 microSD cards (weve tested up to 16GB!).

All the design files (schematicPCB layoutfirmware) are open source released under the CC-SA v3 license and are available through GitHub.

Note: New firmware is available which allows the OpenLog to be used in Arduino 1.0 and also allows for higher baud rates. Check the links below for more information.

Dimensions: 0.16 x 0.6 x 0.75" (4 x 15 x 19mm)

The OpenLog project is housed on GitHub! Please see the wiki for the most up-to-date firmwarefeature requestsfilesand datasheet specs.

Features:

Log to low-cost microSD FAT16/32 cards up to 16GB
Simple command interface

Edit config.txt file from a computer to change baud rate and other system settings
Three modes:
NewLog creates a new log every power up and immediately starts logging
SeqLog appends a file called  SeqLog.txt  at every power up and immediately starts logging
Command mode starts OpenLog at a command prompt at power up
Configurable baud rates (2400 to 115200bps)
Configure unit through config file or the menu system
Powergroundand RX-I are the minimum connections
Reprogrammable ATmega328 using the Arduino IDE
Two LEDs indicate writing status
Input voltage from 3.3V to 12V
2mA idle6mA at maximum recording rate

# RFID CARD READER/WRITER 13.56MHZ




**Description:**

**RC522 chip:**

MF RC522 is applied to the highly integrated read and write 13.56MHz contactless communication card chipNXP launched by the company for the "table" application of a low-voltagelow-costsmall size of the non-contact card chip to read and writesmart meters and portable handheld devices developed better choice. The MF RC522 use of advanced modulation and demodulation concept completely integrated in all types of 13.56MHz passive contactless communication methods and protocols. 14443A compatible transponder signals. The digital part of to handle the ISO14443A frames and error detection. In additionsupport rapid CRYPTO1 encryption algorithmterminology validation MIFARE products. MFRC522 support MIFARE series of high-speed non-contact communicationtwo-way data transmission rate up to 424kbit/s. As new members of the 13.56MHz reader card series of highly integrated chip familyMF RC522 MF RC500 MF RC530 There are a lot of similaritiesbut also have many of the characteristics and differences. Communication between it and the host SPI mode helps to reduce the connection narrow PCB board volumereduce costs.

**RFID module:**

The MF522-AN module the the original Philips MFRC522 chip design circuit card readereasy to uselow costand applies to the user equipment developmentthe reader and the development of advanced applicationsthe need for the user RF card terminal design/production. This module can be directly loaded into the various reader molds. Utilizes a voltage of 3.3Vthrough the SPI interface simple few lines directly with any user CPU motherboard connected communication can ensure that the module is stable and reliable workdistance card reader;

# NEUROSKY MINDWAVE EEG




Description:

EXERCISE EQUIPMENT FOR THE MIND
Research grade EEG technology for home use
Neuroscience meditationmental fitnessand game applications
Lightweightwireless headset with passive sensors
Remember that episode of Firefly when the crew breaks into an Alliance hospital and Simon puts River in the brain scanning machine? Of courseit's all special effects. But that scene totally made us yearn for some brain scanning technology we could use at home. And in walked MindWavelooking all space-aged and awesome. We'll be in our bunkprogramming some Mindwave games...

The NeuroSky MindWave turns your computer into a private tutor. The headset takes decades of laboratory EEG technology research and puts it in your hands. Experts agree that your brain needs to be exercised just as much as your body. Strap on the headset and the MindWave Education system will monitor the attention level of students as they interact with mathmemoryand pattern recognition applications. Ten apps are includedsome educational and kid-friendly and others are seriously fun for all ages. Developmental tools allow you to write your own programs to interact with the Mindwavetoo.

Product Specifications

Research grade EEG technology for home use
Exercise your mind with specially designed neuroscience meditationmental fitnessand game applications
Developmental tools allow you to write your own programs to interact with the Mindwave
Lightweightwireless headset with passive sensors
eSense Brainwave Patterns measures: AttentionMeditationEye Blink
Neuroscience frequency bands: 0.5 to 50hz
Package includes: MindWave HeadsetWireless USB AdapterBonus Application DiscQuick Start Guide.
10 hours of use on one AAA Battery
System Requirements:
PC: Windows XP or Vista & 7Intel Core 2 Duo or Equivalent1 GB RAMDirectX 9.0+ capable1 GB HD space
Mac: OS X 10.5 “Leopard” or betterAny Intel Mac1 GB RAM1 GB HD space

# ADJUSTABLE DC-DC STEP DOWN POWER SUPPLY MODULE WITH LCD DISPLAY




Description:

This is a DC-DC Step Down Power Supply Adjustable Module With LCD Display, the display using STN liquid crystal display screen, voltage and current can be displayed simultaneously, blue background and white character, wide angle and wide field of view. It start used button to adjust voltage, left button to reduced the voltage, the right button to increasing voltage, press button once can adjust 0.04V voltage, if you hold more than 1s, you can quickly adjust voltage. Calibration method: Under power off situation, Holdind left button and power the supply,when the display begin flashing, release left button, with multimeter measuring the output voltage, by press the left and right button ,adjust the multimeter measuring voltage near 5V, such as 5.00V ,4.98V or 5.02V is ok too,at this situation,please ignore the displaying on this item. After adjustment,please power off it and then power to it again, then the calibration is completed. Errors can be calibrated in 0.04V, if the error is greater, you can try to calibrate it again.

Features:

Converter Type: DC-DC
Power Chip Type: MP2307
Input Voltage: 5~23V (Best voltage 20V)
Output Voltage: 0V~16.5V (Continuously adjustable)
Peak Current: 3A
Mainboard Size: 50mm x 30mm x 12mm(LxWxH)
Display Module: STN LCD
LCD Size: 37.5mm x 17.0mm(LxW)

LCD Accuracy: 1%
Conversion Efficiency: 95%
Load Regulation S(I): <= 0.8%
Voltage Regulation S(U): <= 0.8%

# WIND SPEED SENSOR TTL



Description:

YGC-FS wind sensor structure using conventional three wind cupswind cup selection ABS materialhigh strengthgood start; sophisticated signal processing unit can output various signals according to user needs. This product has great rangegood linearityeasy observationstable and reliableand can be widely used for meteorologicaloceanographicenvironmentalairportsportslaboratoriesindustry and agricultureand transportation and other fields.

Use:
Port near the ground floor level of wind direction and speed measurement
Field of meteorological environment
Airport Expressway Monitoring
Agricultureforestryenvironmental protection industry
Parameters:
Measuring range: 0-70m / s
Accuracy: ± (0.3 + 0.03V) m / s (V: wind speed)
Start wind speed: ≤0.3m / s
Input voltage: 5V
Output signal: Pulse signal output TTL
Working environment: Temperature -20 ° C ~ 60 ° C; humidity ≤100% RH
Features:

Meets WMO World Meteorological Organization specification (CIMO Guide)
High sensitivity: Start wind speed ≤0.3m / s
Measuring range: Measuring range 0-70m / s
High Accuracy: Accuracy is ± (0.3 + 0.03V) m / s (V: wind speed)
For a variety of harsh environmentslarge wind strength

# WIRELESS CHARGER 5V 600MA POWER SUPPLY



**Description:**

The Wireless Charging Module can be applied in electronic equipments in common use for close wireless charging or power supply. Consisting of a transmitter and insulation coil, it could serve as a replacement for the Wireless Power Supply with stable 5V output voltage and maximum 600mA output current. Its small size and insulation coil is more suitable for using in wireless project.

**Features:**

- Input Voltage: 12Vdc
- Input Voltage (limits): 13.5Vdc
- Output Voltage: 5Vdc
- Output Current (maximum): 600mA
- Transmitter Coil Inductance: 30uH
- Transmit-receive distance: 1-20mm
- Transmitter : 17 x 12 x 4mm
- Receiver:  24 x 10 x 3mm
- Coil Diameter: 38mm
- Coil Height: 2mm

- بعض انواع واستخدامات محرك ال DC

# DC MOTOR STANDARD 130



Description:

DC Hobby Motor Micro Motor Toy Motor DC Motor Type 130 for Robotic VEG92 P

Name:130 motor
Axial length:8mm
Diameter of axle:2mm
Specification:20*15*25mm
Rated voltage: 3V
No-load speed: 8000 r / min (max)
Load current: 100mA
Stall Current: 0.7A
Weight: 17 grams
Application:used to connect to the main shaft gear and propeller, model cars, model aircraft

# H16 CW CCW MOTOR



Description:

RC Drone Spare Parts CW/CCW Motor for JJRC H16 RC Quadcopter for Tarantula X6 Accemblage Accessory Parts

-The item is a simple and practical quadcopter
-It is mainly made of durable material for long-term use, and designed to offer your motor great stability and support

Type: Motor
Package weight: 0.112 kg
Package size (L x W x H): 6.00 x 6.00 x 4.80 cm / 2.36 x 2.36 x 1.89 inches
Package Contents: 1 x CW Motor, 1 x CCW Motor

Note:

We do not accept any responsibility or liability for the incorrect purchase of our products. All of the products on the website are extensively tested to comply with rigorous and strict QC standards. For all of our clothing products, items and accessories, please take careful note of the precise measurements, mainly shown in centimeters (CM). As part of our company policy, we will not be able to accept returns or provide refunds for items which are the wrong size.

Features:

- 100% brand new and high quality Replacement for JJRC H16 YiZhan Tarantula X6 RC Quadcopters
- Specially designed for YiZhan Tarantula X6 RC Quadcopters
- Offer your motor great stability and support
- Compact size and durable structure, easy to install and use

# DC 12V DOOR ELECTRIC LOCK ASSEMBLY SOLENOID



Description:

Applied to the cabinet lock,locker locks,file cabinet locks ,luggage locks,electric locks,door locks,solenoid locks,drawer,newspaper boxes lock ,sauna lock,locker electromagnetic locks,electric locks,newspaper boxes,sauna electronics lock
Designed with the open frame type and mount board,high power
Easy to install for the electric door lock or other automatic door lock systems with the mounting board.

Features:

Size: L27mm*W29mm*H18mm
Lead length: 25mm
Locking telescopic length: 10mm
Powered form: interrupted
Unlock time: 1 second
Voltage: 12VDC
Current: 0.35A

# N20 12V 200 RPM HIGH TORQUE ELECTRIC GEAR BOX MOTOR




Description:

N20 motor is Mini 12V DC 200 RPM High Torque Electric Gear Box Motor.
Features:

Voltage range: 1.5-12 V

The motor shaft long:10 mm

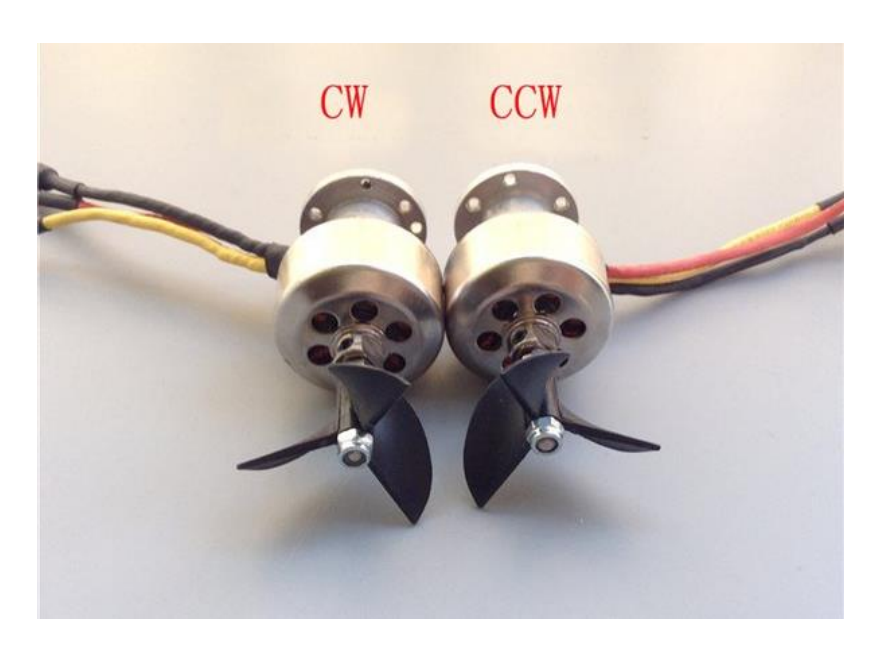
The rated speed:

3.0 V 50 RPM/min

6.0 V 100 RPM/min

12V 200 RPM/min (RPM).

# RC FISHING BOATS 4130 BRUSHLESS MOTOR UNDERWATER + THRUSTER 3-BLADES PROPELLERS




Description:

RC Fishing Boats 4130 Brushless Motor Underwater Thruster 3-blades Propellers Sprayer Pump Thruster for RC Bait Boat Jet Turbine Engine

- Item name:4130 brushless motor underwater thruster
- Thruster length:64mm
- Motor diameter:30mm
- Diameter for nylon propellers:35mm
- Diameter for aluminum mounting seat:25mm
- Thickness for aluminum mounting seat:3mm
- Screw hole:3mm
- Voltage range:12v-24V
- Motor power can be used between 12-24V The higher the voltage, the better
12 -V recommended to use 30A ESC, 24V recommended to use 40A ESC

**Note:**

- The boat uses a single thruster,then it should be uses a positive propeller

- The boat uses double thrusters: using CW CCW propellers, it can cancel each other's interference caused by the rotation of the motor. Let the boat stay in a straight line

# DRILLING DC MOTOR HIGH SPEED LARGE TORQUE



Description:

Rated voltage: 16.8V
Voltage range: 12-19V
Voltage parameters:
When the 12V voltage, current 1.65A, speed 14000rpm / min
When 6V voltage, current 1.25A, speed 4300rpm / min
Motor length: 50mm
Motor diameter: 35.5mm
Stepped diameter: 13mm
Step height: 4mm
Fixed aperture: M3
Fixed Pitch: 25mm
Shaft length: 12mm
Shaft Diameter: 3.175mm
Weight: 165.6 g

# CNC SPINDLE 500W



Description:

CNC Spindle 500w ER11 Collet 52MM Diameter DC Small 0.5kw Air Cooled Spindle Motor for Engraving Milling Spindle with Mount bracket and screw

Operating voltage: 100VDC
Power:500W
Speed:Maximum idling up to 12,000 rev / min
Diameter:52mm
Water Cooling ER11 collet
High precision spindle runout:0.01-0.03mm

# N30 MICRO DC MOTOR SUIT WITH PROPELLER DIY MODEL AIRPLANE AIRCRAFT MAKING



Description:

N30 Micro DC MOTOR Suit with Propeller DIY Model Airplane Aircraft Making

Voltage: 3-6V

Current: under 3.7V, no load current is 0.1A, locked-rotate current is 1A

Speed: 3.7V 17000RPM

Motor shaft diameter: 1mm

Motor size: 20*10*12mm

Motor shaft length: 5mm

Motor weight: about 8.39g

Propeller length: 7.5cm

Propeller hole diameter: 0.95mm (cloth fit 1mm motor shaft).

# MINI SUBMERSIBLE WATER PUMP DC 2.5-6V 120L/H



Description:

Mini Micro Submersible Water Pump DC 2.5-6V Low Noise Brushless Motor Pump120L/H

Specification:

Brand new

DC Voltage: 2.5-6V

Working current: 130-220mA

Power: 0.4-1.5W

Maximum lift: 40-110cm / 15.75"-43.4"

Flow rate: 80-120L/H

Outside diameter of water outlet: approx. 7.5mm / 0.3"

Inside diameter of water outlet: approx. 4.7mm / 0.18"

Diameter: approx. 24mm / 0.95"

Length: approx. 45mm / 1.8"

Height: approx. 33mm / 1.30"

Wire length: about 15-20cm (red: " + ", black(white): " - ")

Material: engineering plastic

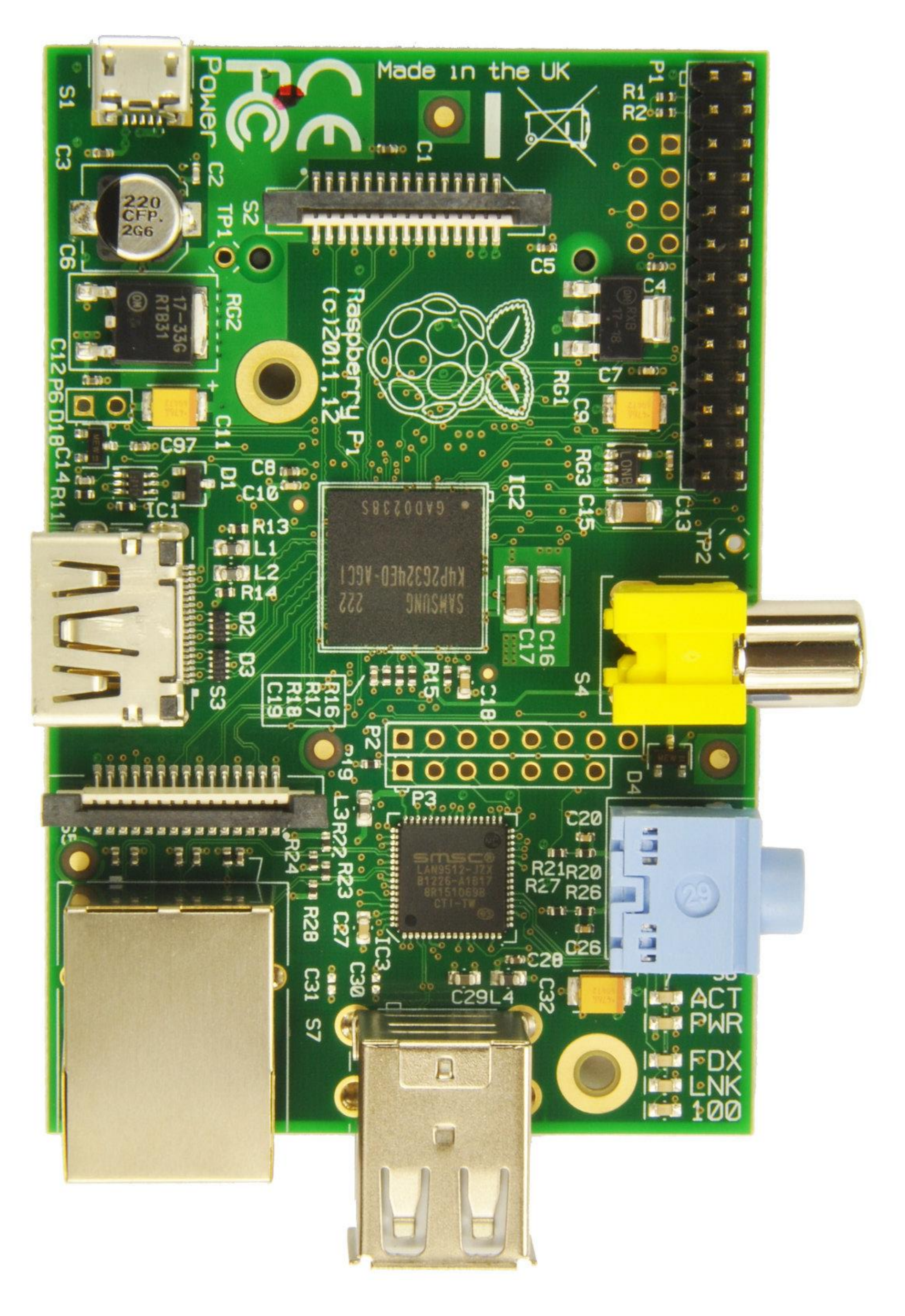
Driving mode: brushless DC design, magnetic driving

Continuous working life of 500 hours

- ملحق B انواع لوحات الراسبيري باي

Raspberry Pi B

وهو اول نوع ظهر عام 2012

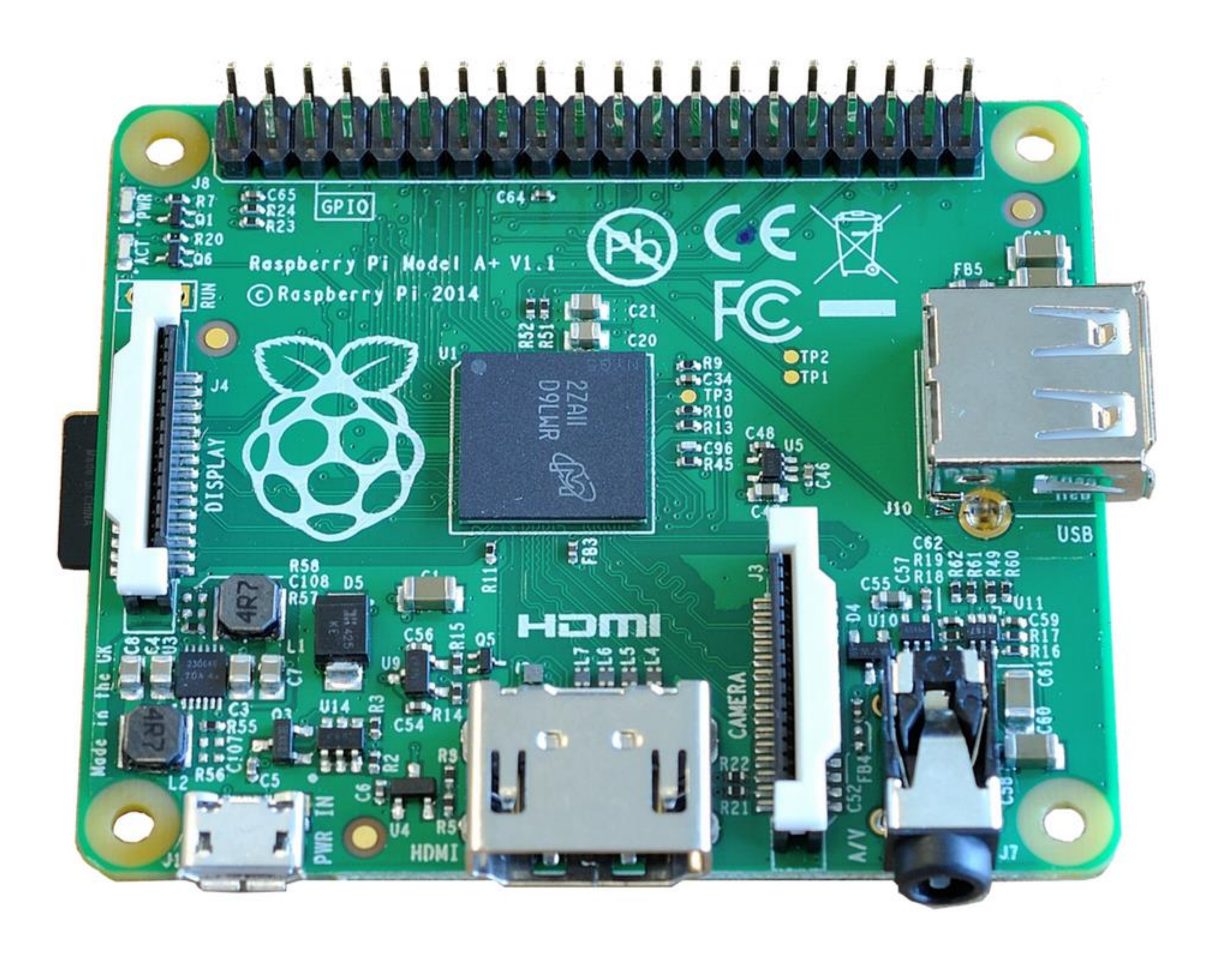


model B mostly w/some model A detail

| | |
|---|---|
| **Release date** | 2012 Feb 15 |
| **Price** | US$35.00 |
| **SOC Type** | Broadcom BCM2835 |
| **Core Type** | ARM1176JZF-S |
| **No. Of Cores** | 1 |
| **GPU** | VideoCore IV 1080p@30 |
| **CPU Clock** | 700 MHz |
| **RAM** | 512 MB |
| **USB Ports** | **Yes** 2x USB 2.0 |
| **Ethernet** | **Yes** |
| **HDMI port** | **Yes** |
| **Analog Video Out** | **Yes** Composite video |
| **Analog Audio Out** | **Yes** 3.5mm jack |
| **Analog Audio In** | **-** via GPIO, USB Mic or USB Sound Card |
| **SPI** | **Yes** |
| **I2C** | **Yes** |

| | |
|---|---|
| **GPIO** | **-** 26-pins |
| **LCD Panel** | **Yes** DSI |
| **Camera** | **Yes** DSI |
| **SD/MMC** | **Yes** SD, SDHC and SDXC up to 2TB |
| **Serial** | **-** Through Expansion Connector, needs level shifting |
| **Wi-Fi** | **No** |
| **Bluetooth®** | **No** |
| **Dimensions** | |
| **Height** | 2.12 in (53.98 mm) |
| **Width** | 3.37 in (85.6 mm) |
| **Depth** | 0.66929 in (17 mm) |
| **Weight** | 1.58 oz (45 g) |
| **Website** | raspberrypi.org |
| **Power ratings** | 700 mA @5V |
| **Power sources** | microUSB or GPIO |
| **Power Over Ethernet** | **No** |

Raspberry Pi A+

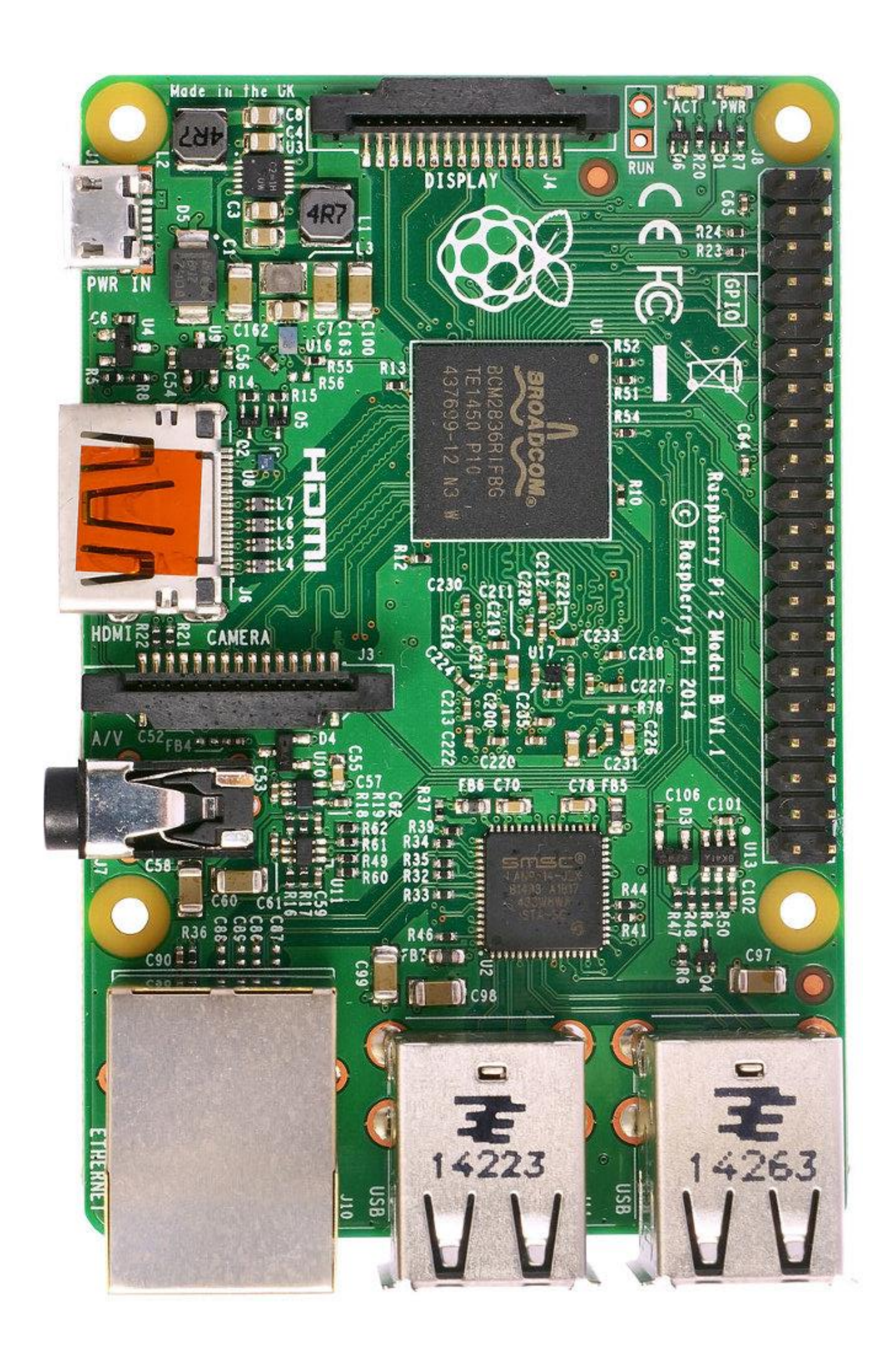



| Release date | 2014 Nov 10 |
|---|---|
| **Price** | US$20.00 |
| **SOC Type** | Broadcom BCM2835 |
| **Core Type** | ARM1176JZF-S |
| **No. Of Cores** | 1 |
| **GPU** | VideoCore IV |

| | |
|---|---|
| **CPU Clock** | 700 MHz |
| **RAM** | 256 MB |
| **USB Ports** | **Yes** 1 |
| **Ethernet** | **No** |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** |
| **Analog Video Out** | **Yes** shared with audio jack |
| **Analog Audio Out** | **Yes** 3.5mm jack |
| **Analog Audio In** | **No** |
| **SPI** | **Yes** |
| **I2C** | **Yes** |
| **GPIO** | **Yes** |
| **LCD Panel** | **Yes** |
| **Camera** | **Yes** |
| **SD/MMC** | **Yes** microSD |
| **Serial** | **-** |
| **Wireless Connectivity (On-Board)** | |

| | |
|---|---|
| **Wi-Fi** | **No** |
| **Bluetooth®** | **No** |
| **Dimensions** | |
| **Height** | 2.55 in (65 mm) |
| **Width** | 2.22 in (56.5 mm) |
| **Depth** | 0.39370 in (10 mm) |
| **Weight** | 0.81130 oz (23 g) |
| **Website** | raspberrypi.org/... |
| **Power ratings** | 200 mA |
| **Power sources** | microUSB or GPIO |
| **Power Over Ethernet** | **No** |

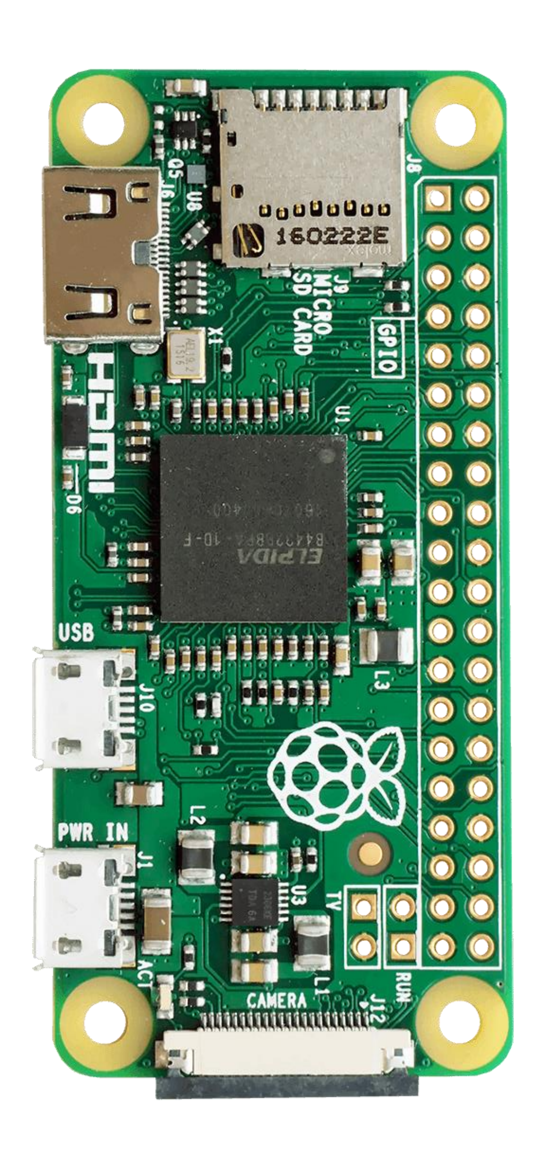
Raspberry Pi 2



| | |
|---|---|
| **Price** | US$35.00 |
| **SOC Type** | Broadcom BCM2836 |
| **Core Type** | Cortex-A7 |
| **No. Of Cores** | 4 |
| **GPU** | VideoCore IV |
| **CPU Clock** | 900 MHz |
| **RAM** | 1 GB |
| **Flash** | |
| **USB Ports** | **Yes** 4 |
| **Ethernet** | **Yes** 10/100M |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** |
| **Analog Video Out** | **Yes** shared with audio jack |
| **Analog Audio Out** | **Yes** 3.5mm jack |
| **Analog Audio In** | **No** |
| **SPI** | **Yes** |
| **I2C** | **Yes** |
| **GPIO** | **Yes** |

| | |
|---|---|
| **CAN** | **No** |
| **LCD Panel** | **Yes** |
| **Camera** | **Yes** |
| **SD/MMC** | **Yes** microSD |
| **Serial** | **-** |
| **Wi-Fi** | **No** |
| **Bluetooth®** | **No** |
| **Dimensions** | |
| **Height** | 3.37 in (85.6 mm) |
| **Width** | 2.22 in (56.5 mm) |
| **Depth** | 0.66929 in (17 mm) |
| **Weight** | 1.58 oz (45 g) |
| **Website** | raspberrypi.org |

Raspberry Pi Zero




| Price | US$5.00 |
|---|---|
| SOC Type | Broadcom BCM2835 |
| Core Type | ARM1176JZF-S |
| No. Of Cores | 1 |

| | |
|---|---|
| **GPU** | VideoCore IV |
| **CPU Clock** | 1 GHz |
| **RAM** | 512 MB |
| **USB Ports** | **Yes** micro + micro OTG |
| **Ethernet** | **No** |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** mini |
| **Analog Video Out** | **Yes** via unpopulated pin |
| **Analog Audio Out** | **-** HDMI audio |
| **Analog Audio In** | **No** |
| **SPI** | **Yes** |
| **I2C** | **Yes** |
| **GPIO** | **Yes** |
| **CAN** | **No** |
| **LCD Panel** | **No** |
| **Camera** | **Yes** latest version include a camera connector |
| **SD/MMC** | **Yes** microSD |

| | |
|---|---|
| **Serial** | **-** |
| **Wi-Fi** | **No** |
| **Bluetooth®** | **No** |
| **Dimensions** | |
| **Height** | 1.18 in (30 mm) |
| **Width** | 2.55 in (65 mm) |
| **Depth** | 0.19685 in (5 mm) |
| **Weight** | 0.31746 oz (9 g) |
| **Website** | raspberrypi.org/... |

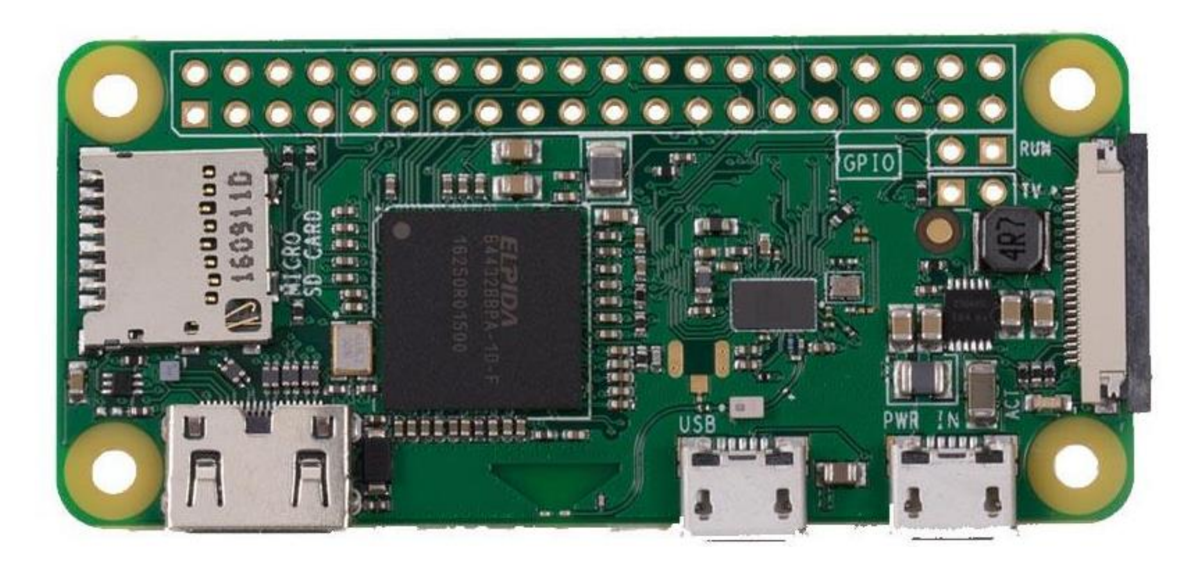
Raspberry Pi 3



| | |
|---|---|
| **Height** | 3.37 in (85.6 mm) |
| **Width** | 2.22 in (56.5 mm) |
| **Weight** | 1.58 oz (45 g) |
| **Price** | US$35.00 |
| **Technical details** | |
| **CPU** | 1.2GHz 64-bit quad-core ARMv8 |
| **GPU** | Broadcom VideoCore IV @ 300 MHz |
| **RAM** | 1 GB DDR2 |
| **Onboard storage** | |
| **Ethernet (LAN, RJ45)** | **Yes** 10/100 |
| **USB** | **Yes** 4x USB2.0 + micro OTG |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** |
| **Wi-Fi** | **Yes** 802.11n |

| | |
|---|---|
| **Bluetooth®** | **Yes** 4.1 LE |
| **RTC** | **-** optional |
| **Additional** | |
| **Release date** | 2016 Feb 29 |

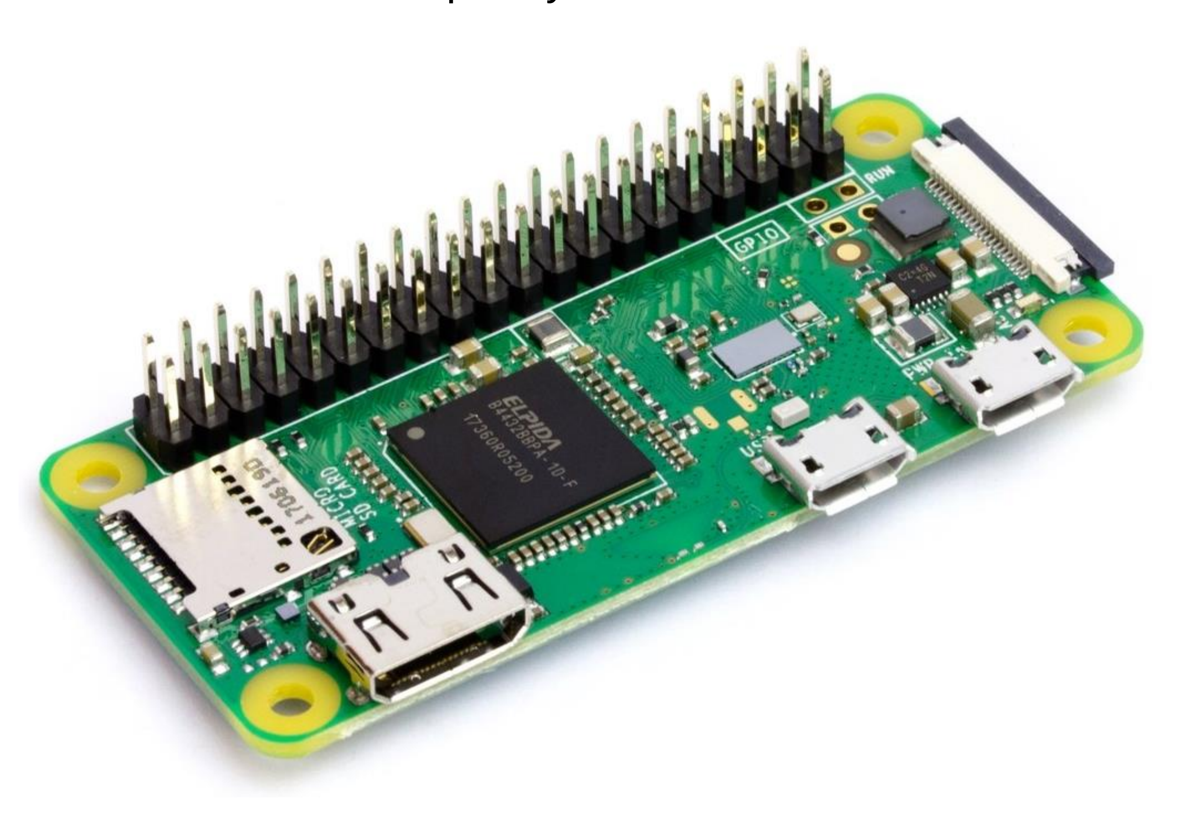
Raspberry Pi Zero W




| | |
|---|---|
| **Height** | 1.18 in (30 mm) |
| **Width** | 2.55 in (65 mm) |
| **Weight** | 0.31746 oz (9 g) |
| **Price** | US$10.00 |
| **Technical details** | |
| **CPU** | 1 GHz Low Power ARM1176JZ-F |
| **GPU** | Dual Core VideoCore IV® Multimedia Co-Processor |
| **RAM** | 512 MB |

| | |
|---|---|
| **Onboard storage** | |
| **Ethernet (LAN, RJ45)** | **No** |
| **USB** | |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** mini |
| **Wi-Fi** | **Yes** 802.11n |
| **Bluetooth®** | **Yes** 4.1 |
| **RTC** | **No** |
| **Additional** | |
| **Release date** | 2017 Feb 28 |

# Raspberry Pi Zero WH

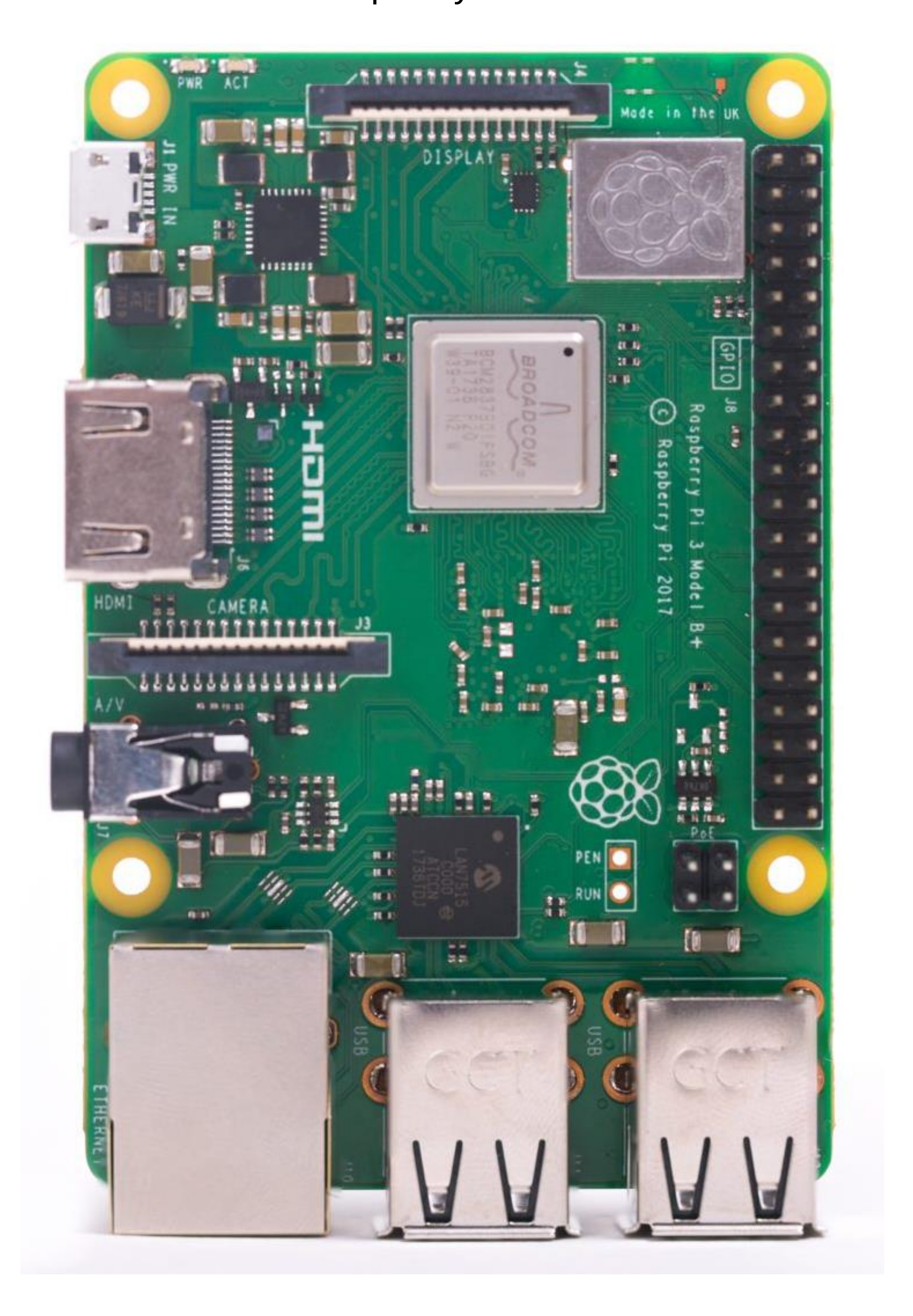



| | |
|---|---|
| **Release date** | 2018 Jan 12 |
| **Product details** | |
| **Price** | US$15.00 |
| **SOC** | |
| **SOC Type** | Broadcom BCM2835 |
| **Core Type** | ARM1176JZF-S |

| | |
|---|---|
| **No. Of Cores** | 1 |
| **GPU** | VideoCore IV |
| **CPU Clock** | 1 GHz |
| **RAM** | 512 MB |
| **Wired Connectivity** | |
| **USB Ports** | **Yes** micro & micro OTG |
| **Ethernet** | **No** |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** mini |
| **Analog Video Out** | **Yes** via unpopulated pin |
| **Analog Audio Out** | **-** HDMI audio |
| **Analog Audio In** | **No** |
| **SPI** | **Yes** |
| **I2C** | **Yes** |
| **GPIO** | **Yes** |
| **LCD Panel** | **No** |
| **Camera** | **Yes** |
| **SD/MMC** | **Yes** microSD |

| | |
|---|---|
| **Serial** | **-** |
| **Wireless Connectivity (On-Board)** | |
| **Wi-Fi** | **Yes** 802.11n |
| **Bluetooth®** | **Yes** 4.1 |
| **Dimensions** | |
| **Height** | 1.18 in (30 mm) |
| **Width** | 2.55 in (65 mm) |
| **Depth** | 0.51181 in (13 mm) |
| **Weight** | 0.42328 oz (12 g) |
| **Website** | raspberrypi.org/... |
| **Power** | |
| **Power ratings** | 180 mA |
| **Power sources** | microUSB or GPIO |
| **Power Over Ethernet** | **No** |

Raspberry Pi 3 B+

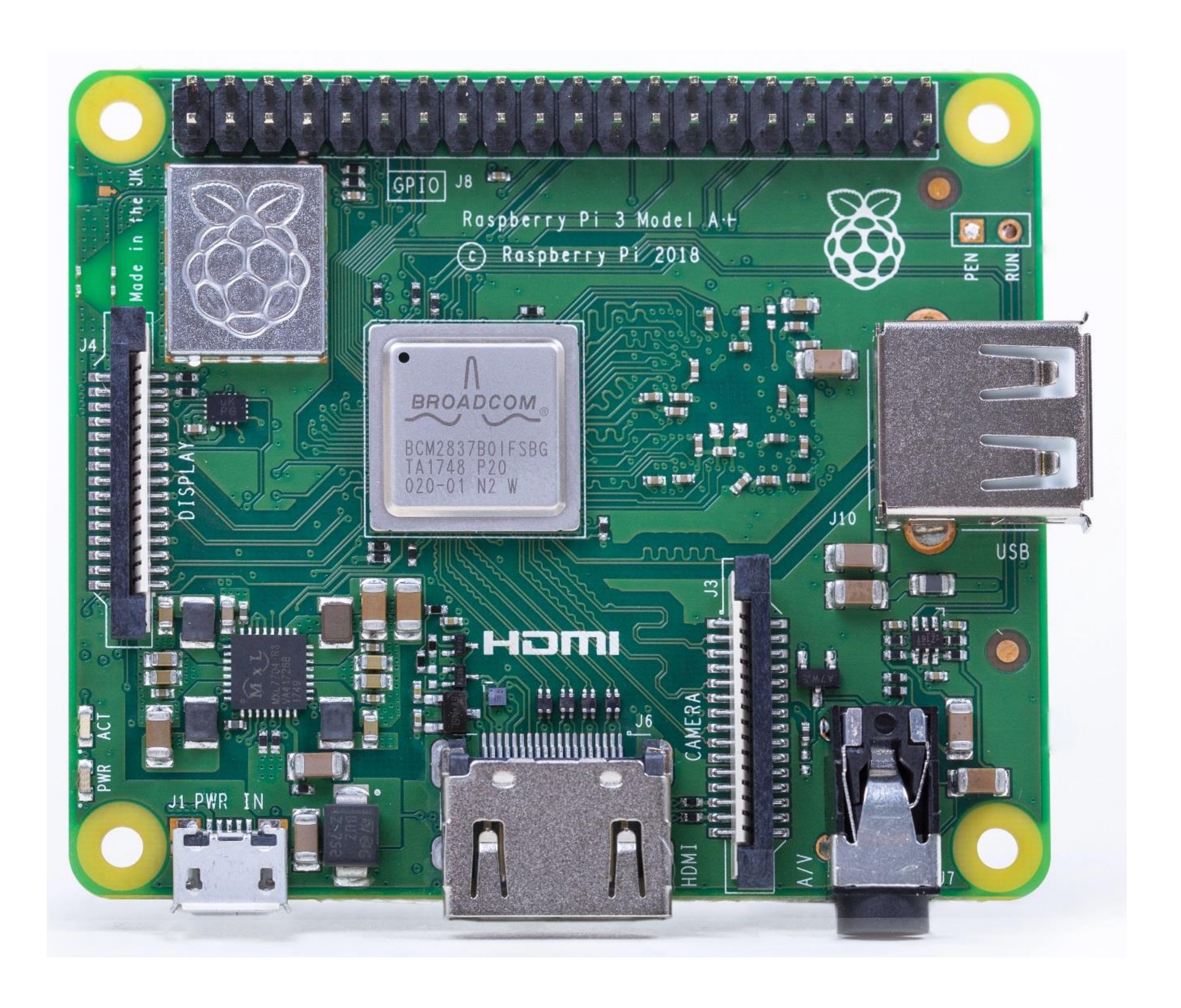


| | |
|---|---|
| **Product details** | |
| **Price** | US$35.00 |
| **SOC** | |
| **SOC Type** | Broadcom BCM2837B0 |
| **Core Type** | Cortex-A53 64-bit |
| **No. Of Cores** | 4 |
| **GPU** | VideoCore IV |
| **CPU Clock** | 1.4 GHz |
| **On Board Memory** | |
| **RAM** | 1 GB DDR2 |
| **Flash** | |
| **Wired Connectivity** | |
| **USB Ports** | **Yes** 4xUSB 2.0 |
| **Ethernet** | **Yes** Gigabit - Over USB 2.0 |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** |

| | |
|---|---|
| **Analog Video Out** | **Yes** shared with audio jack |
| **Analog Audio Out** | **Yes** 3.5mm jack |
| **Analog Audio In** | **No** |
| **SPI** | **Yes** |
| **I2C** | **Yes** |
| **GPIO** | **Yes** 40-pin |
| **CAN** | |
| **LCD Panel** | **Yes** |
| **Camera** | **Yes** |
| **SD/MMC** | **Yes** microSD |
| **Serial** | **-** RX/TX UART |
| **Wireless Connectivity (On-Board)** | |
| **Wi-Fi** | **Yes** 2.4GHz and 5GHz 802.11 b/g/n/ac |
| **Bluetooth®** | **Yes** 4.2, BLE |
| **Dimensions** | |
| **Height** | 3.37 in (85.6 mm) |

| | |
|---|---|
| **Width** | 2.22 in (56.5 mm) |
| **Depth** | 0.66929 in (17 mm) |
| **Weight** | 1.58 oz (45 g) |
| **Website** | raspberrypi.org/... |

Raspberry Pi 3 Model A+




| | |
|---|---|
| **Release date** | 2018 Nov 15 |
| **Product details** | |
| **Price** | US$25.00 |
| **SOC** | |

| | |
|---|---|
| **SOC Type** | Broadcom BCM2837B0 |
| **Core Type** | Cortex-A53 64-bit |
| **No. Of Cores** | 4 |
| **GPU** | VideoCore IV |
| **CPU Clock** | 1.4 GHz |
| **RAM** | 512 MB DDR2 |
| **Wired Connectivity** | |
| **USB Ports** | **Yes** 1xUSB 2.0 |
| **Ethernet** | **No** |
| **SATA Ports** | **No** |
| **HDMI port** | **Yes** |
| **Analog Video Out** | **Yes** shared with audio jack |
| **Analog Audio Out** | **Yes** 3.5mm jack |
| **Analog Audio In** | **No** |
| **SPI** | **Yes** |
| **I2C** | **Yes** |
| **GPIO** | **Yes** 40-pin |
| **LCD Panel** | **Yes** |

| | |
|---|---|
| **Camera** | **Yes** |
| **SD/MMC** | **Yes** microSD |
| **Serial** | **-** |
| **Wireless Connectivity (On-Board)** | |
| **Wi-Fi** | **Yes** 2.4GHz and 5GHz 802.11 b/g/n/ac |
| **Bluetooth®** | **Yes** 4.2, BLE |
| **Dimensions** | |
| **Height** | 2.55 in (65 mm) |
| **Width** | 2.20 in (56 mm) |
| **Depth** | 0.43307 in (11 mm) |
| **Weight** | 1.02 oz (29 g) |
| **Website** | raspberrypi.org/... |
| **Power** | |
| **Power ratings** | |
| **Power sources** | microUSB or GPIO |
| **Power Over Ethernet** | **No** |

تم بحمد الله الانتهاء من الكتاب

انتظرونا في الاصدار القادم



فوزي عبدالله الأزرق
سلسلة الذكاء الاصطناعي
الإصدار الأول
الماتلاب
9
فوزي عبدالله الأزرق
التحكم عن بعد
5
أساسيات في
الإلكترونيات والمستشعرات
2
فوزي عبدالله الأزرق
الذكاء الاصطناعي
البرمجة المتقدمة
10
فوزي عبدالله الأزرق
الروبوتات
7
فوزي عبدالله الأزرق
طائرات التحكم عن بعد
بناء التطبيقات الهندسية
مقدمة في البرمجة
C++
أصيل
Design
0776077988
المتحكمات الإلكترونية
ATMEGA8-16PU
0550I
تصميم مختبر الالكترونيات